



**Australian Government**

**Department of Defence**

Defence Science and  
Technology Organisation



# **An Automatic Transcriber of Meetings Utilising Speech Recognition Technology**

Steve Graham and Jarrah  
Sladek

DSTO-CR-0355

**DISTRIBUTION STATEMENT A**  
Approved for Public Release  
Distribution Unlimited

20040713 107



**Australian Government**  
**Department of Defence**  
Defence Science and  
Technology Organisation

# An Automatic Transcriber of Meetings Utilising Speech Recognition Technology

*Steve Graham and Jarrah Sladek*

**Command and Control Division**  
**Information Sciences Laboratory**

DSTO-CR-0355

## **ABSTRACT**

Conducting meetings is an important part of the Australian Defence Organisation (ADO) activities. Meetings span a broad range of knowledge sharing and collaborative activities; examples are interviews, informal discussions, brain-storming sessions, video-conferences and presentations. Thus, looking at ways to improve the capture of information from these meetings is of interest to the ADO. This report presents progress towards the application of speech recognition technology to automatically produce text transcriptions of collaborative meetings. The report reviews the literature on automatic transcription systems and describes the progress on the research and development of a concept demonstrator named AuTM (Automatic Transcriber of Meetings).

## **RELEASE LIMITATION**

*Approved for public release*

AQ F04-09-1070

*Published by*

*DSTO Information Sciences Laboratory  
PO Box 1500  
Edinburgh South Australia 5111 Australia*

*Telephone: (08) 8259 5555*

*Fax: (08) 8259 6567*

*© Commonwealth of Australia 2004*

*AR-013-054*

*March 2004*

**APPROVED FOR PUBLIC RELEASE**

# An Automatic Transcriber of Meetings Utilising Speech Recognition Technology

## Executive Summary

The aim of this report is to inform the sponsors of the DSTO Task JTW 01/092, Human Computer Interaction Research, including Speech and Natural Language Approaches, for Command Support of the work being conducted in using automatic speech recognition (ASR) technology to automatically capture the discussions in collaborative work efforts. Our sponsors requested this research in an effort to reduce the workload involved in capturing information generated from collaborative efforts such as meetings, brainstorming sessions, interviews or other domains where people need to communicate verbally.

The approach taken was to first develop a concept demonstrator that could demonstrate that it is possible to utilise current ASR technology in producing a useful textual transcript of everything that is said during a collaborative effort. This transcript could then be kept as a full record of the collaboration or be used to produce a summary and formal minutes.

The concept demonstrator used the concept of time stamping each utterance as it is spoken and then interleaving these utterances based on the timestamps. This concept demonstrator was first developed using multiple computers with their clocks synchronised via infra-red (IR) devices. Each computer had a software program that managed this time stamping and a separate program would interleave these utterances at the end of a meeting. The transcript was not available in real time and relied heavily on specially built hardware. A more flexible, portable and less hardware-dependent concept demonstrator was desirable.

In 2001 work commenced on developing a concept demonstrator, now known as AuTM (Automatic Transcriber of Meetings). This system would allow computers to connect via a Local Area Network (LAN) and produce a transcript in near real time utilising ASR technology. The DSTO Technical Report DSTO-TR-1498 describes this early version of the system.

This Client Progress Report focuses on the development of the AuTM project during 2002 and early 2003. It describes work conducted on the development of a more modularised system architecture that will allow for a flexible approach to further development. As part of this modularisation process new components were researched and designed including a component that would allow the system to use any commercial ASR. The work also included combining the separate client and server applications into one and developing a new graphical user interface (GUI). Video and

streaming audio components were added to allow the system to be used in a distributed meeting environment, not just in the one room. As the system developed its usability was considered and automating the meeting logon process, to minimise the information a user would need to provide, was investigated.

In conjunction with the development of the concept demonstrator we investigated what other organisations are doing in the area of computerised meeting room systems and multimedia meeting capture tools. At the time of the investigation there were separate projects being conducted by BBN Technologies, International Computer Science Institute (ICSI) Berkeley University and Interactive Systems Laboratory (ISL) Carnegie Mellon University. All three projects used automatic speech recognition technology but focused on developing their own in-house speaker-independent ASR applications as opposed to commercially available speaker-dependent ASRs as used in AuTM. These projects focused strictly on the meeting room environment where AuTM is designed to be a more flexible and portable system.

Future work for the project is also proposed in this report including the evaluation of the current system, further improvements to the video display and audio steaming, use of third party messaging systems, operation on multiple platforms, transcript storage and retrieval, and integration with language translation systems.

Continuation of the project will depend on future resources and the further development of ASR technology. As this technology improves so will the accuracy and usability of the AuTM system.

The results from the work carried out by the AuTM project team will assist DSTO and the task sponsors in utilising ASR technology to capture and produce textual transcripts of meetings and collaborative efforts. This research will also keep the Australian Defence Organisation informed on what is happening in the area of computerised meeting room systems and multimedia meeting capture tools.

## Authors

### **Steve Graham**

Command and Control Division

*Steve Graham graduated with a B. Information Technology in the School of Informatics and Engineering from the Flinders University of South Australia in 2001. In 2002 he undertook a Graduate Diploma in Business Enterprise with the University of Adelaide whilst working at DSTO on the AuTM project. Steve had a successful fifteen year career with the Department of Employment, Education, Training And Youth Affairs before returning to study to 1998. His interests include integration of speech technology, information capture storage and retrieval, and software configuration management.*

---

### **Jarrah Sladek**

Command and Control Division

*Jarrah Sladek graduated with a B.Comp.Sc. in the School of Mathematics and Computer Science from the Adelaide University in 2001. In 2002 he completed a Graduate Diploma in Business Enterprise at the University of Adelaide, whilst undertaking placement with DSTO as part of the Graduate Industry Linked Entrepreneurial Scholarship. In 2003 he completed a Master in Project Management whilst undertaking full-time work at DSTO. His areas of interest include intelligent environments, computer supported collaborative work tools, speech recognition and speech integration.*

---

# Contents

1. INTRODUCTION.....	1
2. RESEARCH INTO COLLABORATIVE TECHNOLOGIES.....	2
2.1 Computerised Meeting Room Systems.....	2
2.2 Multimedia Meeting Capture Systems.....	4
2.3 Other Commercial Activities.....	4
2.4 Summary of research findings.....	5
3. AUTM PROJECT BACKGROUND.....	7
3.1 Automated Text Extraction of Meetings.....	7
3.2 Automatic Transcriber of Meetings.....	8
3.3 Field Voice Recognition System.....	8
4. SUMMARY OF WORK.....	9
4.1 Overview.....	9
4.2 System Architecture.....	9
4.2.1 Modularisation.....	9
4.2.2 Combining the Client and Server into One Application.....	12
4.2.3 Improving Source Code Readability.....	13
4.3 Multiple Speech Recognition Engines.....	14
4.3.1 Microsoft Speech API.....	14
4.3.2 SAPI Compliant Third Party Components.....	14
4.3.3 Comparison of SREs With SAPIs.....	16
4.3.4 Automatic Speech Recognition (ASR) Interface Component.....	16
4.4 Graphical User Interface.....	17
4.5 Video Streaming.....	20
4.6 Audio Issues.....	23
4.6.1 Audio Recording.....	24
4.6.2 Audio Streaming.....	27
4.7 System Connectivity.....	28
4.7.1 Automating the Connection Between Client and Server.....	28
4.7.2 Upgrade of Sockets.....	30
5. CONCLUSIONS.....	32
6. FUTURE WORK.....	33
6.1 Evaluation of the AuTM Prototype.....	33
6.2 Video display and Recording.....	35
6.3 Improving Connectivity.....	35
6.4 Use of Third Party Messaging Systems.....	35
6.5 AuTM on Multiple Platforms.....	35
6.6 Transcript Storage and Retrieval.....	36
6.7 Language Translation.....	36
7. ACKNOWLEDGEMENTS.....	37
8. REFERENCES.....	37

<b>APPENDIX A:</b>	<b>ASSESSMENT OF VIDEO CODECS.....</b>	<b>40</b>
	<b>A.1. Aim:.....</b>	<b>40</b>
	<b>A.2. Background:.....</b>	<b>40</b>
	<b>A.3. Method:.....</b>	<b>40</b>
	<b>A.4. Results: .....</b>	<b>40</b>
<b>APPENDIX B:</b>	<b>TESTING THE WINDOWS 98 VERSION OF WAVECLONE.....</b>	<b>42</b>
	<b>B.1. Overview.....</b>	<b>42</b>
	<b>B.2. Aims .....</b>	<b>42</b>
	<b>B.3. Methodology .....</b>	<b>42</b>
	<b>B.4. Results .....</b>	<b>43</b>
<b>APPENDIX C:</b>	<b>BRIEFING PAPER ON AUTM'S UTILISATION OF ELVIN .....</b>	<b>44</b>
	<b>C.1. Introduction.....</b>	<b>44</b>
	<b>C.2. What is Elvin?.....</b>	<b>44</b>
	<b>C.3. How could AuTM use Elvin? .....</b>	<b>44</b>
	<b>C.4. Possible Advantages .....</b>	<b>45</b>
	<b>C.5. Possible Disadvantages.....</b>	<b>45</b>

## Acronyms

API	Application Programming Interface
ASR	Automatic Speech Recognition
AuTM	Automatic Transcriber of Meetings
COTS	Commercial of the Shelf
CSCW	Computer Supported Collaborative Work
EMS	Electronic Meeting Systems
FVRS	Field Voice Recognition System
GUI	Graphical User Interface
LAN	Local Area Network
NS or DNS	Dragon NaturallySpeaking
PDA	Personal Digital Assistant
SAPI	Speech API
SDK	Software Development Kit
SMAPI	Speech Manager API
SRE	Speech Recognition Engine
SSL	Secure Socket Layer
TATF	Time and Text File program
TETES	Text Extractor of Time Embedded Speech program
TCP/IP	Internet Protocol
VAC	Virtual Audio Cable
VCL	Delphi Visual Component Library
IDE	Integrated Development Environment

# 1. Introduction

This report provides information to the clients of DSTO Task JTW 01/092, Human Computer Interaction Research, including Speech and Natural Language Approaches, for Command Support. The report is written in response to the milestone: *Application of language technology for speech-to-text transcription.*

A pertinent application of speech recognition technology is automatic speech-to-text transcription during collaborative work within headquarters. Recording minutes of a meeting, an interview, or automatically taking notes from a brainstorming session into a database can be very efficient and cost-effective in terms of human resource requirements. This concept has generated considerable interest in the Australian Defence Force (ADF).

In response to this interest the Speech and Natural Language Task has conducted considerable research and development in this area of speech-to-text transcription during collaborative work. A prototype of an automatic transcriber of meetings known as AuTM has been developed. It uses the Dragon NaturallySpeaking (NS) automatic speech recogniser (ASR) to produce a transcription of a meeting or collaborative work effort. A detailed technical description of this prototype can be found in Zschorn, et al (2003).

There were two major aims in continuing the AuTM project. Firstly, continue the research in using ASRs to automatically produce records of meetings. Secondly, develop further improvements to the AuTM software prototype. These improvements are to advance the software from a concept demonstrator to a stage where external contractors can develop the software for production. This software could then be used by the ADF to automatically transcribe meetings and produce reliable accurate records.

To gain an overall picture of what is occurring in the area of automating collaborative work environments, section 2 details the investigation of what research and development other organizations are doing in this area and how it compares to the work with AuTM. Considerable research is being conducted mainly in large US organizations and universities. Section 3 provides some background on the development of AuTM from its original conception and a brief description of its features. The details of the work carried out during 2002 and the first half of 2003 are included in section 4. Section 5 provides some conclusions on the work conducted during this period. Finally Section 6 briefly describes intended future work for the AuTM project to improve the prototype further.

## 2. Research into Collaborative Technologies

This section details the research work that relates to AuTM in the main areas of computerised meeting room systems and multimedia meeting capture tools. It also details information on commercially available products that are similar to AuTM.

### 2.1 Computerised Meeting Room Systems

Previous work outlined by Zschorn et al (2003) has indicated there are at least three projects related to AuTM. These are being conducted at:

- BBN Technologies (Colbath et al, 1998),
- International Computer Science Institute (ICSI) Berkeley University (Janin 2001, Morgan et al 2001), and
- Interactive Systems Laboratory (ISL) Carnegie Mellon (Waibel et al 1998, 2001; Yu et al 1998, 1999 and 2000; Gross et al 2000).

While these projects are all very similar to each other in their approach, they differ significantly from *AuTM*'s approach.

Each project uses speaker-independent speech recognisers that were initially developed for slightly different purposes and then adapted and optimised for the meeting domain to enable them to recognise the speech of all speakers. This is the most significant difference between these projects and *AuTM*, which uses a commercial off the shelf (COTS) speaker-dependent speech recogniser for each participant in the meeting. The BBN, ICSI, and ISL projects take the approach of developing and optimising their own speaker-independent speech recogniser to capture utterances from all speakers. While this approach means they have far more control over the speech recogniser, these speech recognition systems are less powerful than NS. For instance, NS version 6 has a vocabulary of approximately 250,000 words (Scansoft 2002), while the speech recognisers used in the other projects have between 30,000 and 45,000 words (Yu et al 2000, Colbath et al 1998). This makes out-of-vocabulary (OOV) errors a major problem for the other projects. To help solve the OOV problem the ISL project (Yu et al 2000) uses the Internet to obtain extra vocabulary for the speech recogniser.

The BBN technologies (Colbath et al, 1998) and ISL (Waibel et al, 1998) projects have a strong emphasis on meeting browsers. These are applications used to navigate text, audio and video records of a meeting. The BBN meeting browser is particularly advanced, with time-aligned automatic topic classification, and a search function. In contrast, *AuTM* produces simple HTML or MS Word transcripts. The automated approach of the BBN and ISL systems contrasts with that of *AuTM*, which leaves the summarising to the moderator of the meeting. It is assumed that *AuTM*'s approach is more accurate, but at the expense of operator time.

Waibel et al (2001) mentions using the ISL system with speakers in remote locations. This is clearly where AuTM is heading in the near future, and a task to which it is particularly suited. Although it is not stated explicitly, the other three projects centre on a 'special' meeting room, which may be wired with microphones and computers. In contrast AuTM has a more mobile, distributable nature.

Like AuTM, the ICSI project (Morgan et al 2001) uses head-set microphones, although they also make recordings on lapel and desk microphones in the hope of introducing this less obtrusive technology in the future. ISL's system uses lapel microphones only (Yu et al 2000).

Another related research project is the Meeting Manager work at Massachusetts Institute of Technology (MIT) (Oh et al 2001). It is part of their e21 and Oxygen projects. This project is focused on aiding the facilitator work and compiling summaries of meetings. The work centres on recording short movie clips of important sections of a meeting, rather than using a speech recogniser to put the discussion into text. The facilitator is responsible for identifying which parts of the discussion need to be recorded, and they can label and annotate those movie clips with text information using various methods, including speech recognition. The facilitator is responsible for formulating detailed agenda information. At the conclusion of the meeting this information is included with the movie clips and entered into a SQL (*Structured Query Language*) database. So, like AuTM, the MIT project is reliant on a human moderator to input intelligent annotation and summaries of the discussion.

Much work has been conducted on Electronic Meeting Systems (EMS) (Nunamaker et al 1991). These systems typically consist of 10-20 personal computers linked together in special meeting rooms running custom software. The user interface for the meeting tools is based on text/keyboard input. The software often organises meetings by dividing proceedings into three phases: idea generation (brainstorming), organisation (grouping), and prioritisation (voting). The meeting software needs to be configured before the meeting starts. After the configuration is completed, participants are required to follow a predefined procedure and organisation of the group process. This allows less room for informal interactions and ways of capturing these interactions. No connection to a publicly available interactive workspace or display is supported.

EMS has been shown to improve the quality of group decisions and the time needed to reach agreements (Nunamaker et al 1991). However, the hardware needed to run these systems makes them prohibitively expensive and impractical to use in many settings. Furthermore, these systems may force the focus of meetings to document creation, redirect some of the group's attention to complex computer interfaces, or require participants to type during meetings, which can be disruptive (Landay et al 1999). AuTM was specifically designed without relying on custom hardware. That is, the system uses a COTS ASR and desktop or laptop PCs. Systems such as EMS are better suited for certain types of structured meetings, such as those focused on decision-

making or idea generation. In contrast AuTM imposes less meeting structure and supports a wide variety of meeting styles.

Meeting rooms equipped with special equipment are expensive, so AuTM addresses the fundamental problems of other systems: cost and lack of ubiquity. The AuTM system can basically be used wherever a laptop and network connection exists, which in today's environment poses minimal restrictions. There are a number of advantages in using COTS software. It is freely accessible to users, available at a cheaper cost when purchasing in larger volumes, and readily integrated with other COTS products.

## 2.2 Multimedia Meeting Capture Systems

Multimedia notes and records of meetings provide many benefits over traditional paper counterparts. A video recording of a meeting allows people to review a meeting that they have attended or to catch up on a meeting that they have missed (Chiu et al 1999).

An example of a multimedia meeting capture system is *LiteMinutes*. The system assumes one person is taking the notes, and those notes are typed into an applet on a wireless laptop (Chiu et al 2001). After the meeting, the notes are parsed and a file with the slides and video correlated by the time to each note is emailed to all participants. Users can revise the notes in their email editors and send them back to the server, thus updating the notes displayed on a common web page. Notes, slides and video can also be accessed during a meeting for "instant replays".

Another multimedia meeting capture system is called *NoteLook*, which is a client server note taking system designed to run on tablet computers (Chiu et al 1999). Users can take ink notes, add thumbnail images, annotate video streams, place slides or images on the background of a page, and then view the note files created on a web page. An interesting feature is the ability to summarise notes, by taking snapshots regularly and capturing all slides viewed from all sources.

## 2.3 Other Commercial Activities

Based on extensive research, conducted on the Internet, into commercial bodies developing speech recognition applications, there appears to be no commercial activity, which replicates the exact functions of the AuTM system. However, there are a number of systems that are similar in architecture but are not used in a meeting environment.

Dictaphone Corporation has produced a number of products that are based on the principle of distributive speech recognition with multi-user voice input stations (Kuhnen et al 1999). In particular *ExSpeech* is a system that involves a number of voice input stations and document review work-stations connected to a central dictation

system running on a server computer (Dictaphone, 2002). These voice input stations could be either handheld microphones or headsets connected to networked computers. Speaker recognition capabilities are provided at each voice input station and any authorised user may use the station. The e-mail system of the computer network is used to transport voice files from the voice-input stations to the central transcription system. This system has been deployed in large hospitals to allow medical professionals to dictate and retrieve patient case notes.

The idea of a number of distributed clients communicating to a central server is very similar to the AuTM prototype. It is unclear how the audio signals are converted over the network. Unlike the Dictaphone system outlined, AuTM does not incorporate document review work-stations (Kuhnen et al 1999) to carry out error correction duties. AuTM is considered to be far more portable in terms of error correction strategies and hardware components needed for operation.

IBM Corporation is also a company that has invested heavily in the area of pervasive computing and distributed speech recognition. The *IBM Websphere Voice Server for Transcription* (IBM Corporation(1), 2002) is a specialised speech recognition product that is aimed at software developers and services providers. According to IBM 'the product provides transcription (or deferred recognition) functions that can be integrated into a workflow or document management application'. IBM also states that 'multiple users can dictate audio text from various locations and devices such as microphones, handheld recorders and telephones'. The initial market segments that IBM have targeted focus on the transcription services for the medical and legal communities. The work being conducted by IBM has far more in common with Dictaphone Corporation products than AuTM.

Dictaphone and IBM make mention of their products being used in a ubiquitous manner similar to the AuTM system, however, their applications are not specifically designed for group meetings. Furthermore, AuTM provides the ability to annotate the record with functionality such as client/server messaging, and the creation of motion and action items

## 2.4 Summary of research findings

There are a number of research projects and commercial products that share the same goals as AuTM. The four research projects (BBN, MIT, ICSI, ISL) are quite similar, but these have far more in common with each other than any of them have with AuTM. All of these use a single, in-house speaker-independent speech recogniser rather than AuTM's use of a COTS, speaker-dependent, speech recogniser for each meeting participant. AuTM implements a continuous speech recogniser in a manner that increases the capacity of the overall system by offering less specialised operational components. AuTM is intended to be mobile and not rely on specialist hardware, while each of the four projects (BBN, MIT, ISL and ICSI) appear to have been used in

specially set-up meeting rooms. The four projects are older than AuTM, and some have extra, useful features such as automatic summarisation and purpose-built meeting browsers. In practice it seems that all four relevant research projects try to coherently summarise a meeting and make it accessible afterwards, using a timeline to display relevant information, and marking important points of the meeting. AuTM together with the ISL, BBN, and ICSI projects share some characteristics with the MIT project, but what sets the MIT project apart is its use of selectively storing movie clips of meetings, rather than using speech-to-text to generate text transcripts.

The work being conducted at Dictaphone Corporation and IBM uses technology and architecture similar to the AuTM prototype, though they do not focus on the meeting and collaborative work environment.

A scan of the literature has concluded that the most unique features of the AuTM software are: the live development of transcription capabilities created through a computer network, the ability to annotate the transcript during a meeting, the time stamping of utterances and audio recording for correction of utterances. These unique features ensure an accurate record of who said what and when in a meeting transcript and help differentiate AuTM from any other research.

### 3. AuTM Project Background

The AuTM project has been running for three and a half years. It has involved various groups of students working on research in using ASRs to automatically produce records of meetings. The results of this research were then implemented in software to demonstrate the concepts learnt. Three prototypes have been developed, each with a different focus. The first prototype called Automated Text Extraction of Meetings (ATEM) was focused on using Infra Red (IR) devices to synchronise computers during the meeting. This prototype proved the concept of using ASRs in automatically transcribing meetings but was not practical because of the extra hardware required. The second prototype was focused on automatically producing a transcript only using software and removing the need for specialised hardware. AuTM was developed from this prototype. The third prototype developed was focused on automatically producing transcripts of interviews rather than meetings. This prototype was produced for a specific need. These three prototypes are described in this section.

#### 3.1 Automated Text Extraction of Meetings

In 2000, a group of final year Electrical and Electronic Engineering students from the University of Adelaide developed an Automated Text Extraction of Meetings (ATEM). The ATEM project was an integrated system capable of recording the conversation at a meeting and converting those utterances into a transcript of the meeting. This has been achieved through the design and fabrication of four pieces of hardware and the design and development of two independent software programs.

The hardware comprises an IR transmitter module and three IR receiver modules, each of which is connected to a computer via a serial port. These modules provide a means for the computer clocks to synchronise so that all computers at the meeting have access to a synchronised time.

A Time and Text File program (TATF) connects to the NS speech engine, which allows the spoken utterances to be transcribed. The synchronised time is then used to time stamp the text at the beginning and end of each speaker's utterance. The speaker's name is also stamped on each line of text and at the end of the meeting the individual text transcripts are saved for further processing. The TATF software is required to be installed on the computers of each of the meeting participants.

At the completion of the meeting, all the time stamped text transcripts of the individual speakers are collected and uploaded to a computer, which contains the Text Extractor of Time Embedded Speech program (TETES). In TETES the number of speakers involved in the meeting and the associated text files are selected. The program then

aligns the text, so the dialogue appears in a chronological order to provide a complete transcription of the meeting [Krishnamoorthy, 2000].

### **3.2 Automatic Transcriber of Meetings**

Following on from the ATEM project, two Graduate Industry Linked Entrepreneurial Scheme (GILES) students from the University of Adelaide developed a Speech-to-text Transcriber for Computer Supported Collaborative Work (CSCW) [Hashemi and Littlefield, 2000], later known as AuTM.

The AuTM prototype uses speech recognition technology to aid transcription of meetings and other collaborative work efforts. The system allows up to eight meeting participants each with a microphone and a computer running an AuTM client program. Each meeting requires one of the participants to run an AuTM moderator program. The AuTM client and moderator programs communicate over a TCP/IP network rather than using IR devices, as is the case in the ATEM prototype. During the meeting each of the AuTM client programs transcribe spoken utterances into text, logs their start and stop times, and saves the audio as wave files. The saving of audio allows for future error correction in the text if required. To allow the spoken utterances to be transcribed and recorded at the same time an audio splitter was required to provide separate audio sources.

The AuTM moderator program allows the meeting moderator to annotate the transcription during the meeting by including an agenda, highlights, motions and actions items. The final transcription can be saved as either an MS Word document for further manipulation, or as a HTML document to be viewed over the Internet. By the end of 2001, the AuTM prototype had reached version 1.3 and was developed using Delphi 5 and the commercial speech recogniser used was NS Professional v 5 [Zschorn et al., 2003]. The client and moderator programs have now been combined into the one program to simplify the system.

### **3.3 Field Voice Recognition System**

The Field Voice Recognition System (FVRS) was developed in 2001 as a result of a request from Land Operations Division for a system that could automatically transcribe interviews in the field. It was specially developed for operations in East Timor in February 2002. It utilises much of the technologies and techniques developed in AuTM but does not include the meeting annotation features. It includes the ability to prerecord interviews for later transcription using a mini-disc player. Because this system was to be used on laptops in the field, keyboard commands were included. The system was deployed with extensive documentation and help files. Although the development of this system was not continued in 2002 some of the technologies used have been employed in the further development of the AuTM project.

## 4. Summary of Work

### 4.1 Overview

During 2002 the project team looked at a number of areas that would progress the development of the software and further the aims of the project. These areas included:

- Describing, documenting and developing the system architecture of the application.
- Developing components or modules that could effectively plug into the application with little development work required.
- Improving the Graphical User Interface to make it easier for users to interact with the application.
- Including streaming video and audio to allow the system to be used in a distributed environment and not just in a single room.
- Researching the elimination of extra hardware needed to split the audio signal from a single microphone.
- Combining the two separate server and client applications into the one application so that a user can act as a meeting moderator or meeting participant depending on their role in a meeting.
- Researching other work being conducted with other CSCW tools.
- Researching the automation of the network connectivity of each computer involved in an AuTM meeting. This includes accessing and setting IP addresses.
- Researching the use of third party messaging systems that are being used within DSTO such as Elvin.

Two GILES students, and other students on short-term and work experience placements contributed to this work. The work undertaken culminated in the release of version 1.5.2 of the system.

### 4.2 System Architecture

#### 4.2.1 Modularisation

When the AuTM system was originally developed in 2001 the main push of the project was to quickly develop a concept prototype to show that using automatic speech recognition technology could be used in a collaborative environment for automatically producing a transcript of the discussions. To enable this to happen a Rapid Application Development process was adopted by the developers to provide a concept demonstrator as quickly as possible. Due to this rapid development the overall architecture of the system was not fully considered. During 2002 much thought went into the system architecture of AuTM. A number of meetings were held with

stakeholders in the project to decide on the future architecture. The following aspects were considered:

- What functionality is required and what features should be included to obtain this functionality?
- Could the system be broken up into semi autonomous modules that are loosely coupled together? This could then give the system a more plug-and-play approach that would allow using different components to achieve certain functionality.

From these meetings three modes of operation were considered. These included:

1. A system for meetings being conducted in the one conference room where all participants, including the meeting moderator, would have their own computer and microphone.
2. A system where there is only one computer but there is a microphone for each participant.
3. A system for meetings where some participants would be in a separate room and therefore not be able to directly talk to the other meeting participants.

Each operational mode would require the system to have some different functionality. It was decided that all these operational modes should be explored depending on the availability of resources.

The first operational mode was most closely aligned with AuTM version 1.3 so development continued on this system to make it more robust and user friendly. Aspects of the other modes such as multiple microphones on the one computer were investigated but further research will be carried out in the future. Some features of the distributed mode where participants are in different locations were explored. These features included video and audio streaming; refer to sections 4.5 and 4.6 for more information on the research and implementation of these features.

The team decided that the project would benefit from attempting to modularise the system so it would not have to rely on particular third party components or software applications. Any newly developed or third party components could then be easily plugged into AuTM with the minimum of programming effort. For example, the user could select from a range of speech recognisers that they may have on their computer and not have to rely on any particular one. Or the system could use any protocol such as SSL (Securer Socket Layer) for enhanced security rather than TCP/IP. This modularisation concept was kept in mind when further developing the components and features of AuTM. To help developers with this concept of modularisation high-level architectural diagrams of the system were constructed.

Figure 1 is a simple block diagram of AuTM 1.3.4 and is the first attempt at modelling the system. This model illustrates the components of the system and how these components fit together. It helped the developers understand the overall system structure and was useful in explaining the system to others interested in the project.

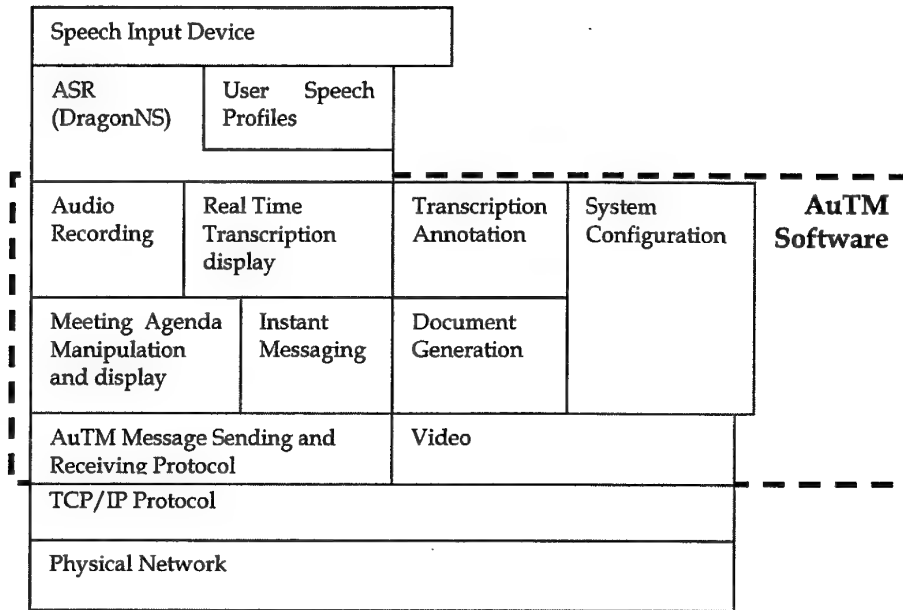


Figure 1 Architecture Diagram of AuTM 1.3.4

As work on the software continued a more sophisticated model was needed to show the connections between the various software components and concepts. To organize this a new model using a layered approach was constructed. Figure 2 illustrates this model showing the AuTM modules and the other components these modules interact with.

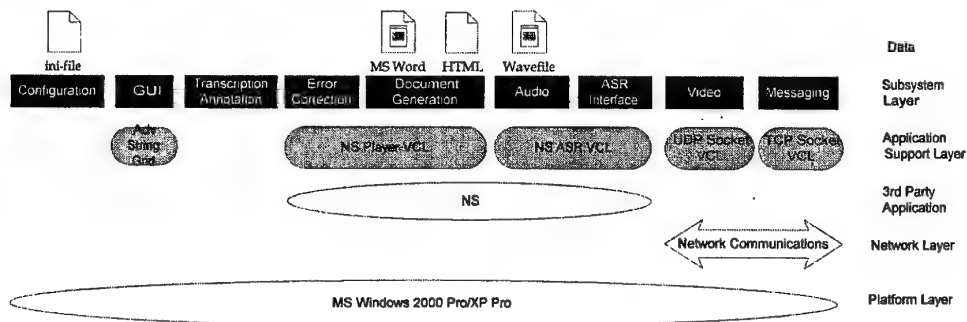


Figure 2 Layered architecture diagram for AuTM 1.5

The Data layer shows various formats of data that AuTM uses or stores during its operation. It uses an ini file to store and retrieve the configuration information for the application. All audio data is stored in wav files, which get created by the clients then transferred to the moderator application for playback when correcting the transcript. Later versions of AuTM used the NS .dra format. The final transcript is then stored in

HTML or MS Word format for future editing or viewing by other applications. The Subsystem layer defines the various modules of the AuTM system similar to those illustrated in Figure 1. The Application Support Layer shows components contained in the Delphi Visual Component Library, which provide tools to allow AuTM modules to access external applications or the Windows operating system environment. The 3<sup>rd</sup> Party Application layer currently includes NS but may contain other applications in the future. Because AuTM uses a network to communicate between computers this layer has been included. The Platform layer describes the operating systems on which AuTM will run.

During the AuTM project, version control numbering, in the form XX.XX.XX, was used to name each version of the system. As there does not seem to be an agreed industry standard to identify what constitutes a number change, the following protocol was used. The third level identifies minor fixes of bugs and cosmetic changes to the GUI. The second level number changed when extra functionality was included and the first level was when major architectural changes were made.

#### 4.2.2 Combining the Client and Server into One Application

The architectural model developed for AuTM was based on a client/server model where there were separate client and server applications to perform different tasks. The client application used NS to convert speech into text, time stamped these utterances then sent them to the server application. The client application also displayed information received from the server, such as the utterances, agenda information, meeting participant information, annotations and video. The server acted as an application for routing the information received from one client to the rest of the clients in the meeting. It also provided the same displays as the clients. Although it could display information to the user, the user would also have to run a client application if they wanted to participate in the meeting. This arrangement was fine for a concept demonstrator but is considered to be overly complicated for a real application. To simplify AuTM and make it more user-friendly the two applications were combined into one.

When combining the client and server the assumption was made that all participants in a meeting will want to contribute to the meeting and that their utterances should be recorded, transcribed and included as part of the meeting record. It was assumed that AuTM would mainly be used in an environment where a meeting moderator would contribute to the meeting and look after advancing the agenda and adding annotations to the transcript as the meeting progressed.

Taking a client and then adding the server functionality, as an option, when the program starts up, was the approach taken when combining the AuTM client and server into the one application. Depending on which option the user requests, the application would either start as a straight client, with an ASR to convert speech into

text, or the application would be both a client and a server. The user who chooses to act as the meeting moderator will have the extra functionality of setting up and advancing the agenda, adding annotations to the transcript and broadcasting messages to all participants. They are also responsible for ending the session and saving the transcript.

In the process of combining the client and server a number of issues were encountered. One major difficulty was the issue of setting the focus for the ASR as NS will direct its output to whichever component has focus. For AuTM to operate properly the focus needs to be on the DictEdit component box which is where NS directs its output after an utterance is spoken. A component obtains the focus when a user clicks on it with the mouse pointer. For example, if the AuTM application is set up to be the meeting moderator, the focus will be set to the agenda component when the user clicks on an agenda item to advance the agenda. The effect of this is that transcription no longer works. To deal with this problem the focus was reset to the DictEdit box each time it loses focus. This issue was not such a problem in the pre-combined version, as the user didn't need to interact with the program via the mouse apart from turning the microphone on and off.

The other major issue was the arrangement of the program source code, which had to be revised to facilitate the new combined functionality. In revising this code procedures and functions were arranged into logical groupings and order. This process also included the underlying concept of modularising the code. This included creating some new units and encapsulating some code from the Main unit into these new units.

#### 4.2.3 Improving Source Code Readability

The main reasons for reducing the size of the source code in AuTM's Main form was to increase the readability of the code and work towards modularisation of the application. Two extra units were created to encapsulate some code from the Main form. The MainTypes unit encapsulated all the variables, classes and functions that were considered global and could be used by other units apart from the Main form. This will help as AuTM becomes more modularised in the future. The other unit created, called MessagingUtils, encapsulates the messaging subsystem used by an AuTM client. Prior to combining the client and server the code relating to messaging was a complex series of 'if else' statements that were effective but difficult to read and understand. This unit uses layers of procedures to encapsulate this functionality, which makes the code easier to read and therefore debug.

### 4.3 Multiple Speech Recognition Engines

As part of the modularization thrust of the AuTM project and to make the system more flexible it was decided to investigate the use of multiple speech recognition engines (SREs). A user could then select the speech recognizer they wish to use from a range of speech recognizers they have available on their computer. The team first investigated the Microsoft Speech API then some products that utilise this technology. We also compared SREs with various development interfaces to see if the concept of selecting multiple SREs is possible to implement in AuTM. In anticipation of this being plausible a separate speech engine component was developed.

#### 4.3.1 Microsoft Speech API

Microsoft has developed their own Speech Recognition Engine (MSRE) and a Speech Application Programming Interface (SAPI) to access this engine. We were motivated to investigate the use of these technologies, for AuTM, as they are free to obtain and free to re-distribute with any application. Currently there are two versions of the SAPI we are interested in. These are versions 4 and 5.1. The older version 4 supports the NS SRE so theoretically we could have AuTM accessing either the NS SRE or the MSRE through the use of the SAPI. Unfortunately the SAPI does not provide the high level components that are provided by the NS Software Development Kit (SDK). The NS SDK uses ActiveX components to access this high level functionality. NS takes care of saving audio, controlling speakers, audio devices etc. They also provide a microphone button that looks after input volume levels, and an edit box for the text being produced by the engine. The SAPI components provide none of this functionality that is provided in NS (Blaney,2002).

#### 4.3.2 SAPI Compliant Third Party Components

We also investigated other companies that were providing similar components to those we were using with NS. These products had to be SAPI compliant to allow us to use either of the two compliant engines we had available, i.e. the NS speech recognition engine and the MSRE. The IBM ViaVoice engine is also available for development, but this uses its own interface called the Speech Manager API (SMAPI). It is unlikely ViaVoice would ever support SAPI since it is a competing technology.

The commercial components that exist for SAPI are wrappers on those provided by Microsoft through their SAPI library. They all start with the basic engine controls mentioned above and introduce the extra functionality such as microphone buttons and speaker controls. For this reason, it would be possible to write our own components based on the SAPI components. It is intended to include this as future work for 2003.

The first commercial components looked at were produced by a company called Wizzard Software ([www.wizzardsoftware.com](http://www.wizzardsoftware.com)). They produce two sets of components, one that implements SAPI and another that implements SMAPI. Both are available as a free download trial version. Because Wizzard allows access to both the SAPI and the SMAPI, this product looked promising. Unfortunately we quickly discovered a problem in its compatibility with Delphi. Apparently a bug exists in Delphi that will not allow the components to install properly. This product should be looked at again when future versions of Delphi are released.

The next commercial component was called DTalk, made by O&A Productions ([www.o2a.com/dtalk.htm](http://www.o2a.com/dtalk.htm)). They provide some useful components that are at a similar level to the NS components currently used in AuTM. They offer a standard speech recogniser component, similar to that provided in SAPI and the NS SDK. DTalk also provides several other useful components that can create dialogue boxes, which allow the user to choose between the available SAPI compliant speech recognition engines and the audio devices on the computer. Using the DTalk components, we attempted to implement the ability to access both the NS and the Microsoft speech recognition engine within the same program. There are some issues that will need to be dealt with if implementing this arrangement in AuTM.

The first is that NS saves audio files to the .dra file format and MSRE uses the .wav format, these are very different and incompatible formats. NS provides a microphone button component that currently provides two important functions in AuTM. It provides the ability to pause and resume the engine from processing audio, and provides a volume input level display. DTalk does not provide components with this functionality so they would have to be built using SAPI components. It may be easier to build separate NS and DTalk programs than try to combine them into one.

A third commercial component was SpeechKit 4 produced by Chant ([www.chant.net](http://www.chant.net)). It provides a large set of components that access SAPI 5.1. We were able to implement the components but discovered an issue that could not be resolved. AuTM is designed to automatically start and stop recording and transcribing an utterance when a speaker starts and stops talking without them having to manually interact with the application. The system does not record the silence between utterances as it only starts recording and transcribing when the speaker's voice is above a certain level. Unfortunately the current version of SpeechKit requires the user to click on a button to start and stop recording an utterance. SpeechKit can record the entire session for a speaker but cannot record individual utterances. This behaviour is not what is required as AuTM needs to record individual utterances to allow them to be interleaved with other meeting participant's utterances.

### 4.3.3 Comparison of SREs With SAPIs

In trying to find a solution for having multiple ASRs available for an AuTM meeting we compared the compatibility of various SREs with certain interfaces that can be used in developing an ASR for AuTM. Table 1 shows the results of these comparisons and indicates that Chant's SpeechKit is the only product that will provide access to all versions of the three speech recognition engines investigated. Unfortunately this product is still under development and, as explained earlier, currently does not provide all the features we require

*Table 1 Comparison of Speech Recognition Engines with Development Interfaces*

	NS	NS	NS	ViaVoice	MSRE	MSRE	DEPHI
	5	6.1	7	10	4	5.1	7
NS SDK 5	✓	✗	✗	✗	✗	✗	✓
NS SDK 6.1	✗	✓	✗	✗	✗	✗	✓
SAPI 4	✓	✓	✓	✗	✓	✗	✓
SAPI 5	✗	✗	✗	✗	✗	✓	✓
D Talk 3	✓	✓	?	✗	✓	✗	✓
Wizzard	✗	✗	✗	✓	✗	✓	✗
SpeechKit 4	✓	✓	✓	✓	✓	✓	✓

Investigation will continue on this matter of trying to accommodate multiple ASRs in AuTM. In anticipation of finding a solution and to further develop the modularisation of AuTM, a new ASR interface component was developed.

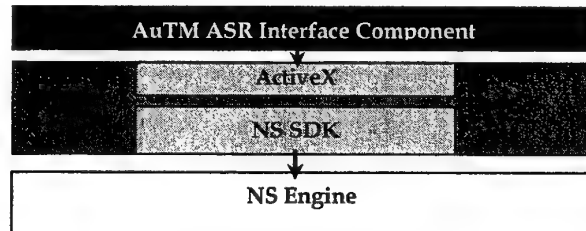
### 4.3.4 Automatic Speech Recognition (ASR) Interface Component

The first stage of introducing multiple ASRs was to separate the speech recognition procedures from the main form. It was decided to create a separate code unit that could then encapsulate the speech recognition processes. This separate unit could then interface with the API of any speech recognition engine. The system would allow the user to enter the name of the recogniser at the start of a meeting and meeting participants would not need to all be using the same recogniser. This Engine Interface component was set up to handle NS, IBM ViaVoice and MSRE. Only NS has been implemented so far as it has proven more difficult than originally thought to include other recognisers.

AuTM accesses the functionality of the NS recognition engine through the use of ActiveX components. The system uses the DgnEngineControl, DgnMicBtn and DgnDictEdit; components which are part of the NS Software Development Kit (SDK). The NS SDK is a component of the Delphi Visual Component Library (VCL). Figure 3 illustrates the connections between these components.

The new interface unit sets up pointers to these NS SDK components, registers the components with the Windows Registry and handles destroying these components

when the program is closed. The unit also handles displaying the dialogue boxes associated with creating, enrolling and changing user profiles plus setting and checking audio input devices. The unit displays messages to the user when there is a problem with any of these dialogue boxes. The procedures in this unit are accessed as the program starts, for initializing the components, and then during the meeting session as the participant uses the microphone button and speaks.



*Figure 3 Architecture diagram for current implementation for the interface between AuTM and NS using ActiveX controls.*

#### 4.4 Graphical User Interface

The AuTM software relies on a Graphical User Interface (GUI) to provide some interaction between the user and the software. Part of the development of the AuTM system involved changing and optimising the existing user interface developed by Zschorn et al (2003). Previously, the GUI developed in the early stage of the project was based on a number of separate windows for each visual component. These windows included a main form identifying the meeting participants, agenda, transcription and highlights. These were all displayed through the main menu. As can be seen in figure 4, each of these windows could be made visible independently of the others, at any time while the program was running. Refer to Zschorn et al (2003) for more detail on the old GUI. Although this was a very flexible architecture for the GUI, controlling the positioning of the windows on screen was often a problem when different size monitors or screen resolutions were used. The use of multiple windows, which could be opened and closed during the meeting, caused some confusion with new users. It was also time consuming to position the windows on the screen. Furthermore, changes to the interface were needed, in order to reflect new additions to the AuTM system such as video.

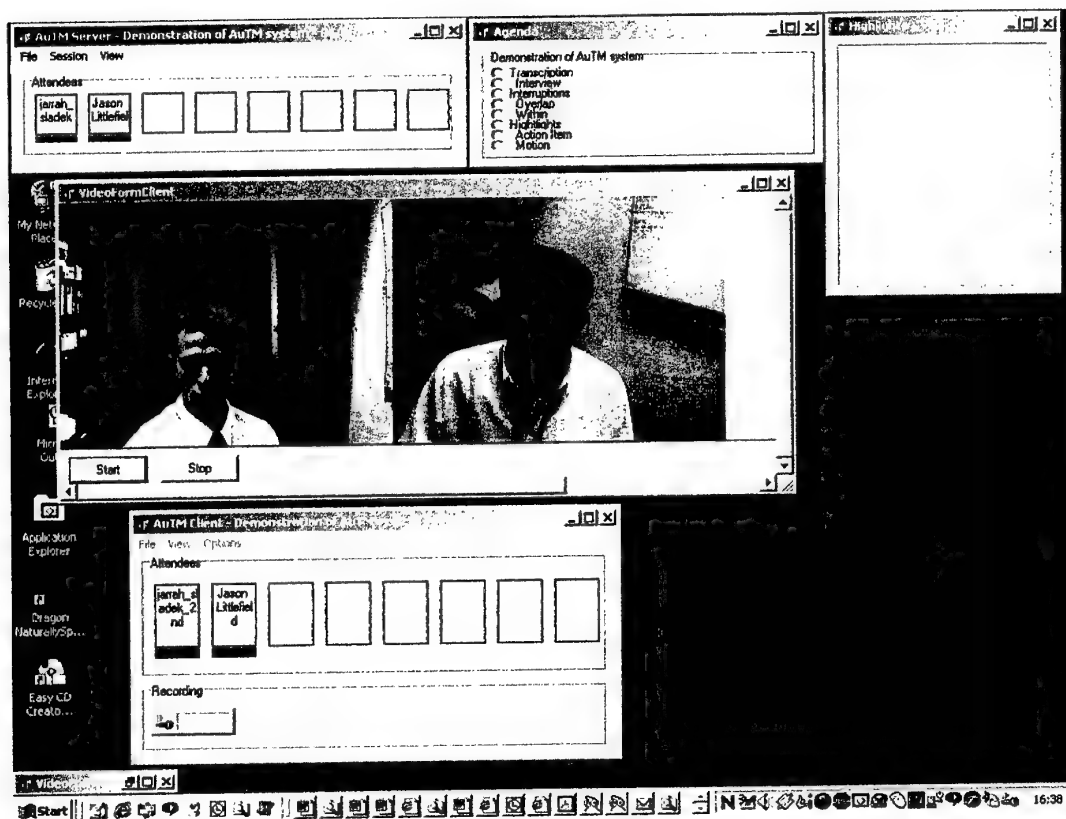


Figure 4 A screen shot of a previous version of AuTM.

The development team decided to change this GUI architecture from multiple windows to one window with multiple frames. Each frame would contain a visual component. A number of GUI experts at DSTO were consulted when deciding on the final GUI architecture. We wanted the GUI to be as usable as possible and ensure proper visual design principles were followed. We consider version 1.5 of AuTM represents a more ideal user interface that would have much more real world use. Figure 5 shows the new single, symmetric client/server program.

The menu structure for the program provides access to functions relating to control of the transcript, meeting session, video imagery and help facility. As the program is designed for the Western hemisphere, where the eye moves from left to right when reading, these critical functions have been placed in the top left hand corner of the GUI. This will enable the user to spot them quickly.

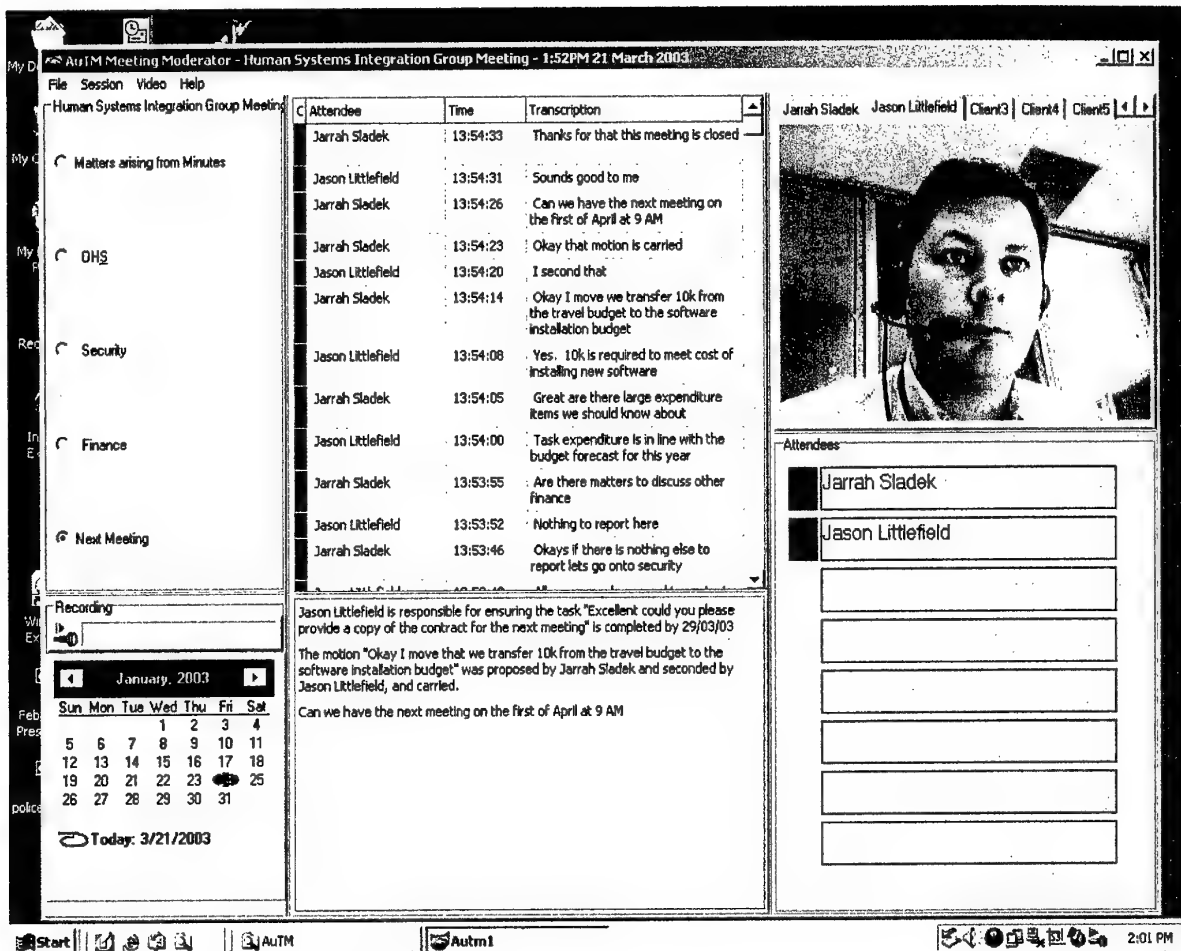


Figure 5.5 AuTM User Interface

The agenda frame occupies the next most important screen area of the GUI to reflect the importance of the agenda during a meeting. The microphone button has been placed near the calendar and agenda frame, as these components will be manipulated during the meeting using the mouse pointer. Keeping the agenda and microphone button closely positioned enables the user to control their contribution of spoken utterances without getting too side tracked by other graphical elements.

The real-time transcription frame is positioned in the middle of the GUI and features quite prominently in the design. The position of the video frame occupies a less important area on the right of screen. By offsetting its position it was hoped that the user could focus attention not just on video, but agenda and transcription as well. Other less prominent functions of the AuTM system include the highlights and user awareness frames, which have subsequently been placed in the lower half of the GUI as these visual components need only be glanced at periodically during a meeting.

## 4.5 Video Streaming

In real-life communication, speakers are largely dependent on both auditory and visual information, including speaker's facial expressions, posture, movements, and appearance (Nazikian et al 2002). According to Mehrabian (1971) as much as 93% of total meaning of a message can result from the non-verbal, visual cues, and only a small percentage is attributed to verbal communication. Whilst it is difficult to gauge an accurate rating of the importance of non-verbal communication, there is little doubt that it is a very important area in face-to-face communication.

In the case of a distributed meeting where speakers are located in different meeting rooms, the streaming of video and audio offers a means of aiding both verbal and non-verbal communication. For example, such functions as identifying and verifying a speaker are more difficult without visualisation of their face.

Implementation of the AuTM video module uses Video For Windows Application Programming Interface to carry out the function. There are two steps involved in streaming video across a TCP/IP network:

- (1) *Video capture*- This capability is handled by using a Microsoft windows message-based interface to access video acquisition hardware. This stage involves connecting the capture window to the camera (capture driver). Each camera capture driver can provide up to four dialog boxes to control aspects of video digitisation and capture process, and define compression attributes used in reducing the size of the video data. Video steaming involves capturing the video frame locally then sending each frame remotely across the network.
- (2) *Video Compression Manager (VCM)* - Provides a means of compressing and decompressing the video data such that it can be easily transported across the network. In general, an application uses VCM to perform the following tasks:
  - a. Locate, open, or install a codec (COmpression/DECompression algorithm).
  - b. Configure or obtain information about the compressor or decompressor
  - c. Use a series of functions to compress, decompress, or display the data to screen.

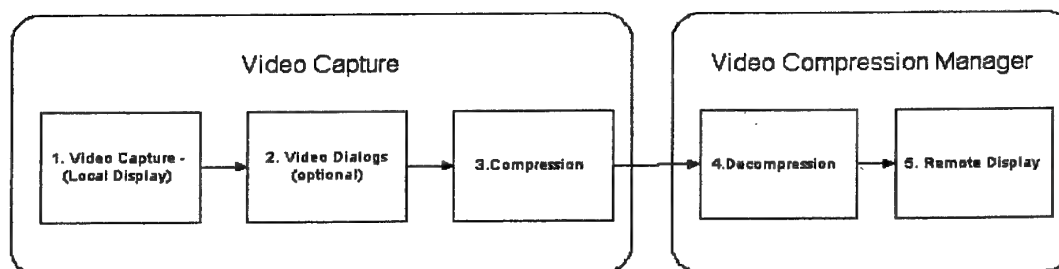


Figure 6 Flow of a general video sequence using video for windows.

A previous version of the AuTM system shown in Figure 4 was designed such that the video could be viewed optionally from the client main menu, consisting of a floating non-maximised window. It was anticipated that the amount of on-screen clutter generated from opening all the windows from the main window would be confusing to the user. Hence this provided one of the main motivations to incorporate the GUI into a single window as described in section 4.4.

Compression was an important factor that determined the layout of the captured video. Digital video needs to be compressed because it takes up a large amount of disk size and bandwidth in its original form. Video compression essentially makes it easier to store and distribute digital video amongst a network.

According to Cisco systems, a market leader in IP video conferencing products, video has a very high and extremely variable packet rate, with a higher than average transmission rate across a network such as TCP/IP.

An analysis of video compression was carried out to determine the average packet size and compression ratios of various codecs that were available. The results of these comparisons are detailed in Appendix C. The results showed that the uncompressed packet size was 230 Kbps. Whilst compression ratios varied greatly between codecs, on average packet size was reduced in the order of 20 times, approximately equating to 9Kbps.

Most personal computer video capture and editing systems capture video using Motion JPEG video compression (MJPEG). In motion JPEG, each video frame is compressed separately using JPEG still image compression standard. MJPEG is a *lossy* compression technique, which means there is some loss in video data as a result of compression and decompression. Testing revealed that a MPEG 3 codec such as Morgan Multimedia (<http://www.morgan-multimedia.com/>) was best suited to the AuTM application. This was mainly due to the variable settings that optimised the display of streamed video across the network. These settings also gave the user the ability to control the quality of the video, by:

- increasing the quality, which increases the packet size and hence time to compress/decompress the frame sent across the network, and
- decreasing the quality, which decreases the packet size and transmission time of packets sent through the network.

Unfortunately this codec is not free to distribute and so has not been included in the current version of AuTM. Future versions of AuTM may be packaged with proprietary licensed codecs such as Morgan Multimedia in order to optimise video display.

Bandwidth was another important factor that determined the resolution and screen layout of each captured video. According to Microsoft the following factors are found

to affect the amount of network bandwidth that audio and video conferencing scenarios use:

- (1) The amount of motion for video; more motion equals higher bandwidth.
- (2) The size of the video window — small, medium, or large.
- (3) The quality of the video — faster video or better quality.
- (4) The type of camera used — black-and-white or colour. The more central processor power the camera uses, the slower the performance, and the less bandwidth used.
- (5) The microprocessor type and speed, and the network speed.

As AuTM utilises the Microsoft Video For Windows Application Programming Interface there are several video dialog boxes, which the systems developer can use to control various aspects of the video digitisation and capture process. These dialog boxes include:

*Video Source* – This provides a selection of input video channels and parameters affecting the video image being digitised in the frame buffer. Such functions as hue, contrast, and saturation of the video are controllable.

*Video Format* – This controls selection of the digitised video frame dimensions and image-depth, and compression options of the captured video.

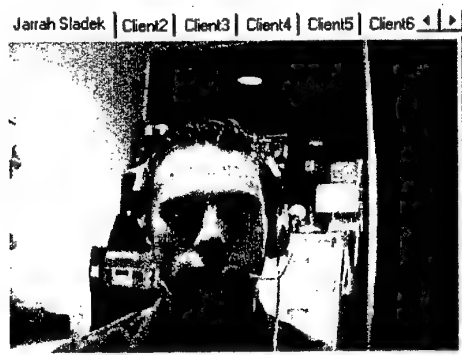
*Video Display* – This controls the appearance of the video on the monitor during capture. The controls in this dialog box have no effect on the digitised video data, but they might affect the presentation of the digitised signal.

*Video Compression (video Codec)* – This is used to open an installed Compressor/Decompressor algorithm to be used for the streaming of video, and control the video frames rate.

In order to shield users from the complexities of video streaming, only the *video format* and *video compression* dialog boxes are made available at run-time through the AuTM application. A default low bandwidth video codec, *Indeo 5.1*, has been programmatically chosen to carry out compression/decompression duties of video frames. This video codec is installed by default with Windows 95/98/2000/XP, so is free to distribute and gives reasonable quality results. AuTM users can change this default codec. However, testing over a LAN revealed that for video transmission to occur, compression and decompression must be processed by the same installed codec for every user. Implementing the codec this way means that users can immediately select the 'start' and 'stop' menu items and begin video transmission without the need to notify every speaker with which video codec will be chosen for the AuTM session.

As mentioned in section 4.4 one of the problems with the earlier GUI design shown in figure 4 was the amount of screen space taken up by the video. As the AuTM application was designed for a maximum of 8 clients, the practical display of each of these clients' video at a resolution of 320x240 pixels was not possible. Concurrent display of eight video clients could only be made possible if video resolution were halved to 160x120 pixels. It was decided that the decreased resolution of video below 320x240 pixels was no longer effective in aiding in speaker identification and facial expression of remote speakers.

In order to overcome the above design issues, the current video interface of AuTM consists of a tabbed panel, whereby each new tab represents a new client in a meeting, as can be seen in Figure 7. This tab display is very similar to the video interface of version 3.0 of Lotus SameTime (IBM Corporation (2), 2002).



*Figure 7 Screen shot of the current video display*

Participants are made responsible for sending their own video by using the start and stop menu items on the main window. A participant's video is sent via the network to the server, this is then broadcast to all participants in the meeting including the originating participant. The video is then displayed locally.

#### **4.6 Audio Issues**

Audio is an important aspect of the AuTM system as it is used for correcting transcription errors and could be used for a distributed version of the system. This section describes the problems we have encountered with recording the audio for correction, what research was undertaken, and finally, the solution that was implemented and why this solution is not the most desirable.

To enable AuTM to be distributed so that meeting participants can be in separate rooms, it is essential they can hear each others' speech in real time, not just read it on the screen. Streaming audio was developed to allow this but not implemented in AuTM 1.5 because of the problem described below.

#### 4.6.1 Audio Recording

As no ASR is 100% accurate, corrections to utterances will need to be made at the end of each meeting. To enable a user to correct the transcript they need to be able to listen to what was actually said by each participant, so all utterances have to be recorded in real time. The AuTM client records this utterance as a wave file, which is sent to the server program at the end of the meeting. The NS software controls the device that the audio signal is sent to from the microphone. This device is the driver software that controls the sound card or USB audio port. This means that no other software, including AuTM, can access this audio signal for the purpose of recording.

To overcome this situation and allow both NS and AuTM to have a signal for recording, hardware was developed to split the signal so it can be fed into two separate audio devices. These might be two sound cards or two USB ports. This signal splitting is a usable solution to the problem for demonstration software but it is very inconvenient, as the user needs to ensure they have this extra hardware. This is not a desirable long-term solution to the problem as this hardware was specifically built for this purpose and is not commercially available. To avoid the signal splitting hardware, a software solution was required.

In finding a software solution, the team searched for existing potential solutions and identified two possibilities. These included the Virtual Audio Cable (VAC) (Muzychenko(1), 2002) and WaveClone (Muzychenko(2), 2002). Evaluation copies of these products were obtained and assessed.

VAC is a Windows multimedia driver allowing the transfer of audio (wave) streams from one application to another. It creates a pair of Wave In/Out devices for each cable. Any application can send an audio stream to an Out device, and any other application can receive this stream from an In device. All transfers are made digitally, providing no loss of sound quality. (Muzychenko (1), 2002)

If more than one application is sending audio to the VAC, it will mix all streams together. If more than one application is receiving audio from the VAC, it will share the same audio data between all targets. This is the feature that interested the AuTM team when investigating this as a possible solution.

The VAC is useful to record a synthesizing application's audio output in real time, or transferring a sound stream to another application for processing. You can, for example, use two or more software audio generators, synthesizers or sequencers to

produce audio streams sending them to VAC Out, and recording the mixed stream from VAC In using recording software such as Windows Sound Recorder,

At the time of conducting the research on the VAC we did not fully understand how the application worked as a virtual audio device. It appeared from the literature (Muzychenko (1), 2002) that the software could only transfer a sound stream from one application to another. This would not provide a solution to the audio splitting problem as neither NS nor AuTM outputs an audio stream. We have since discovered that coupling another piece of software produced by Muzychenko called an Audio Repeater may provide the solution to the audio splitting problem. This Audio Repeater application can transfer the real-time audio stream from any Wave In port to any Wave Out port. This means that an Audio Repeater can be used with the VAC to feed the audio from a real sound card to the Wave Out port of a virtual cable. Because the virtual cable allows for multiple clients, both NS and AuTM could access the Wave In port of the same virtual cable. Figure 8 illustrates this arrangement with arrows showing the flow of the audio signal. This would then give both applications access to the same audio stream. This solution is yet to be fully tested but has been included in future work on AuTM.

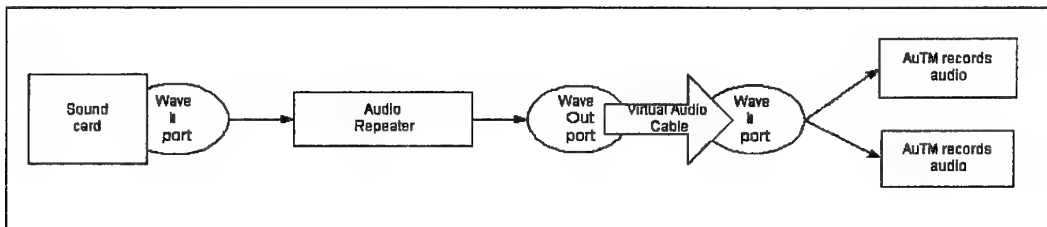


Figure 8 How VAC would be used to connect AuTM to a sound card

WaveClone is an application that effectively allows the user to clone or copy a particular port on an audio device. The clone can then be used in conjunction with the original device to provide an audio stream to both AuTM for recording and to NS for transcribing. The application needs to be registered with Windows on a computer as an audio device. Unfortunately only the demonstration version for Windows 98 was available at the time the research was conducted. AuTM is currently set-up and configured to operate as a concept demonstrator on the Windows 2000 operating system and at present this is the preferred configuration. A demonstration version of WaveClone was tested on a Tecra 8000 laptop running the Windows 98 operating system. Details of the experiment are provided in appendix B. The results showed that this application could provide a simple and acceptable solution to the audio splitting problem if a version was to be developed for the Windows 2000 operating system. With these results in mind we contacted the applications author, Eugene Muzychenko to enquire about access to the source code and to arrange a possible

collaborative development effort. We were informed the source code would cost US\$3,500 and that a Windows 2000 version of the software had been developed and was currently in the testing stage with full release sometime in late 2002. This program is still being pursued.

The team also explored four other techniques to develop a solution. These included:

- (1) Alter AuTM in some way to allow it to take two streams of audio data and feed one into NS and the other into AuTM's TAudio component for recording. Possibly using multithreading within the AuTM application.
- (2) Building an audio driver using the DDK (Windows Driver Development Kit) that could handle multiple outputs.
- (3) Using the Microsoft Speech API to build a solution.
- (4) Using the inbuilt microphone in the Logitech video camera to provide the audio input for AuTM to record.

It was thought that the AuTM application could be altered so that multiple threads of execution could occur after the application had connected to the server. This would allow two procedures to execute at the same time. One procedure could start NS translating the utterance and sending it to the server while the other procedure could use AuTM to record the utterance. This was a naive view of the AuTM system and how multithreading should be used. The idea did not solve the problem of only having one sound source that of course both threads would need to access. In fact, according to Cantù (1999), threads are not worth the trouble if an operation is heavily reliant on external shared data, which AuTM is. Another major problem with using multithreading is that it makes the program run slower which is not desirable in a real time program like AuTM. (Cantù 1999) After researching this idea and gaining a better understanding of threads in Delphi it was decided this would not be a practical solution.

Although building an audio driver using the DDK would be an interesting and worthwhile learning exercise it would only be considered as a practical option if nothing else is available. After researching this technology it was soon realized that the amount of time required to learn the intricacies and complexities of the Windows DDK and then develop a device driver that would solve the problem is well beyond the time and resources available for the AuTM project.

Using the Microsoft Speech API to build a solution was investigated on a number of occasions during the project as more information about the technology came to light. The original investigation showed it would be useful if we were rebuilding AuTM in another language. At this time we did not have a Delphi version of the SDK. SAPI acts as an interface between the speech recognition engine and a user application, in this case NS and AuTM. It provides many audio control functions that an application can

use. It seems to provide many of the services that AuTM currently accesses through the NS SDK. This technology would be useful and will be explored further. More information on the SAPI is included in Section 4.3 Multiple Speech Recognition Engines.

With the recent inclusion of video into the AuTM system a further solution was trialed. Instead of getting the audio signal from the noise cancelling microphones, the inbuilt multidirectional microphone of the Logitech cameras is used to record the utterances. This provides a playback of the speech but with a significant amount of background noise. Although this method does not give high quality playback of the utterances, it is adequate for the purpose of correcting the transcript. Because of the extra noise picked up by the multidirectional microphone it may be difficult or even impossible to clearly hear an utterance in a noisy environment. As AuTM is primarily designed to be used in a single room with video, it would not always be required and therefore no camera would be available for capturing speech for recording. Although this is a possible solution, it is not considered adequate to replace the audio splitting device. It also doesn't meet the aims of the project, which is to provide a software solution to the audio splitting problem.

Unfortunately none of these potential solutions were satisfactory for accessing two separate audio streams. However, in mid 2002, after the research mentioned earlier was conducted, a new version of the NS SDK was released. This SDK was for NS 6.1 and provided some extra functionality that was not available in the NS 5 SDK. The main function that interested the AuTM team was the ability to programmatically save the audio associated with a transcribed utterance using the SessionSave procedure. Having NS save the audio meant that only one stream of audio is required and splitting the audio is no longer needed. There are some serious disadvantages with this solution. The first major issue is that the SessionSave procedure saves the audio in a NS proprietary format, '.dra', and not in the desired more universal Windows Wave format. This means that a special DragonPlayer application is needed to play back this audio when correcting the transcription. DragonPlayer comes as a component that can be embedded in a HTML or Word document. It requires a separate file to be created by AuTM that lists all the audio files to be played. DragonPlayer then uses this list to concatenate separate utterances into a phrase. The problem with this arrangement is that we want the audio to be played back at any time after the meeting. This will not be possible unless DragonPlayer, the file listing the .dra files and the transcript are available in certain directories on the Moderator's computer.

#### 4.6.2 Audio Streaming

The current implementation of audio streaming uses a software development toolkit component called MMTools 2.5 for Delphi. This toolkit is based upon the audio compression manager offered in the Microsoft Windows API. The audio streaming of each client is handled independently from the video streaming. The audio streaming essentially works by initialising an audio stream, which is relayed to the server. The

role of the server is then to merely broadcast this audio stream to all other attached clients (not including the original sender). This feature has been integrated and tested with version 1.5 of AuTM on a 10Mbps LAN. The results have been promising with no voice delay present using the application, and there appears to be no degradation of the quality of output audio.

The development of this audio streaming component has also suffered from the problem of splitting the input audio signal described in section 4.6.1. Because of this, the audio streaming component cannot be integrated into the current version of AuTM 1.5 until this problem is resolved. The main issue is that the same input audio device needs to be used concurrently by audio streaming and the speech recognition engine. As such the speech recogniser will not release this audio device whilst it is in use. Likewise, when audio streaming, the audio input device is locked and cannot be accessed by the speech recogniser. As described in section 4.6.1, AuTM 1.5 has implemented a temporary fix to speech recognition and audio recording functions by using the Dragon SDK to carry out both functions.

Audio streaming is an important area of work for a distributed version of AuTM that needs further attention. There are three possible solutions to this problem. The first solution is to record to a '.wav' file first and then feed it into as many applications as needed. This would result in some delay of real-time transcription and streaming.

A second solution would be to first carry out audio streaming and recording to file, followed by feeding recorded audio into the speech recogniser offline. This solution would not make real-time transcription possible.

A third solution would be to utilise a multi-client audio sharing device driver, which would enable any application to utilise the same input audio device. As pointed out in section 4.6.1 and Appendix B, a commercial product called *WaveClone* exists, however, this product does not work successfully on Windows 2000. Future versions of this product will need to be assessed for the purpose of audio streaming when they are released.

## 4.7 System Connectivity

As AuTM is a networked system that involves several computers communicating with each other, we have looked at improving these connections. We looked at trying to automate the computers finding each other on the network and improving the socket components that are used to connect the separate AuTM applications.

### 4.7.1 Automating the Connection Between Client and Server

In the current version of AuTM (1.5) the user needs to enter the unique Internet address of the computer running AuTM as a meeting moderator. This is known as the

IP address and tells the client application where to look for the meeting moderator application. In making the prototype more useful it is desirable to make this an automatic process so the user connecting to the meeting does not need to know this IP address of the moderator. In late 2002 a work experience student joined the AuTM development team and explored the possibility of automatically detecting the IP address of the computer that is running AuTM as the meeting moderator.

AuTM can run in two different networked environments. It can be run in an open LAN or in a closed LAN environment. An open LAN is a situation where a meeting would take place over a large LAN (several hundred computers), a WAN, or over the Internet. In this environment the IP address is assigned to the computer dynamically as it logs on to the network. In a closed LAN computers are connected directly or via a switch and the IP address is assigned statically by the user in the Properties section of the Windows Network and Dialup Connections dialog. When looking at automating the IP connection process both of these environment scenarios were considered.

In the open LAN it was possible to get a list of all the computers connected to the network and then attempt to connect to each to see if an AuTM server was running. Just within the DSTO network it took eight minutes to return a list of over 1000 computers. This is obviously an unacceptable time delay and so this method was not pursued. It appears that in an open LAN environment it is not possible to automate the connection process, as the client cannot automatically find out the IP address of the moderator's computer. This means that the user running AuTM as a meeting moderator will have to manually tell all meeting participants the computer's IP address. The participants will have to enter this IP address in a dialog box as they log into a meeting. Future work will investigate having AuTM get and display the IP address of the moderator's computer. This could then be displayed to the moderator on the AuTM GUI, instead of them having to manually find the IP address through other means.

In a closed LAN the IP address of each computer needs to be set manually by the user before they connect. In the current version of AuTM there is no assistance to help the user carry out this operation, which requires a level of knowledge beyond most users. Ideally AuTM would automatically set its computers IP address so the user does not need to carry out this operation. A possible implementation of this arrangement is that it be done programmatically by AuTM when it starts up. The AuTM application starting as the moderator would set their computer's IP address to a fixed address such as 10.0.0.1. The first AuTM client to log into the meeting could set their computer's IP address to 10.0.0.2 with each subsequent client checking the availability of this address and then counting up until an address is available. This implementation would further advance AuTM to a usable prototype and is intended to be included in future developments.

#### 4.7.2 Upgrade of Sockets

To extend AuTM's usefulness, in early 2003, the system was tested in various environments, including a wireless LAN at the University of South Australia. Prior to this AuTM had only ever been used over the DSTO wired LAN. In this test on a wireless LAN we encountered problems with clients dropping out during a meeting with the program displaying socket failure error messages. Sockets are components that provide the software level connection between the client and the server applications. This then led us to research socket components other than the standard Delphi sockets being used. The new socket components selected are more sophisticated and allowed us to make the messaging process more efficient. This was achieved by only needing one socket that can handle two-way sending and receiving of messages, rather than two sockets as in the previous version of AuTM. This arrangement reduces the socket connection time and complexity required. Zschorn et al (2002) explains how the dual sockets arrangement worked in AuTM.

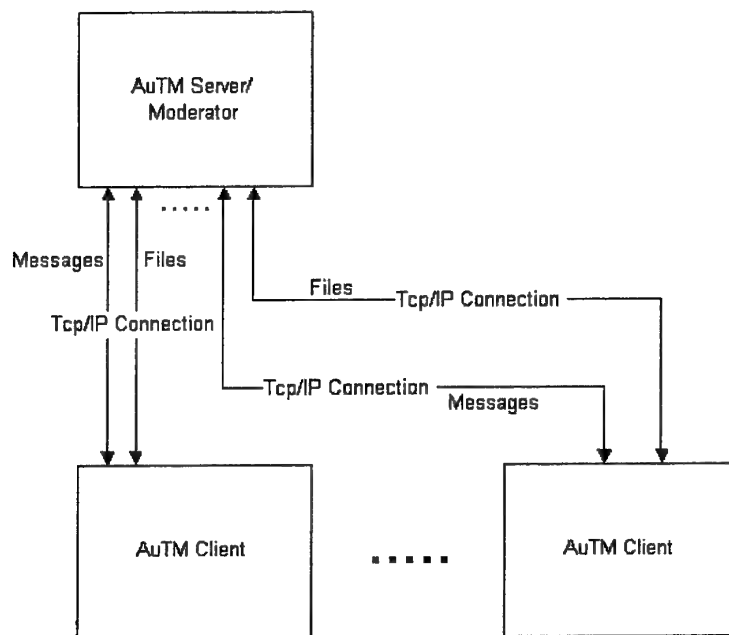


Figure 9: New AuTM Socket Structure

The implementation of these sockets gave us the opportunity to change how AuTM deals with audio files. The files are now sent to the server on a separate socket in a binary block rather than being converted to text and sent as a stream of text. This method is more efficient and quicker than the old method, as the complex set up messages used to tell the server an audio file is being sent, are no longer required.

Figure 9 illustrates the two-way communication between the server and two AuTM clients using separate sockets for text and audio.

Soon after the implementation of these new sockets, we thought about how each AuTM meeting participant had to remain connected to the server application for the entire meeting to enable their audio to be uploaded at the end of the meeting. This situation was not desirable as meeting participants often have to leave a meeting before it ends. The new sockets allowed us to implement a change in the order of AuTMs processes. Audio files can be uploaded from the client's computer to the moderator's computer during the meeting after each utterance is spoken, rather than waiting until the end of the meeting. This enables participants to leave early and have AuTM still capture their audio files for correction at the end of the meeting.

## 5. Conclusions

AuTM has progressively developed from an idea to a system that demonstrates the concept of using automatic speech recognition to aid in the capturing of verbal interaction in collaborative work. In 2002 and 2003 the focus has been on improving the usability of the system and making it more robust and flexible.

Attempting to modularise the system has helped to identify the components that make up the system. This will ensure the system is more flexible allowing a variety of components to be included as they are developed. Combining the client and server programs into one has made the system simpler to use and develop. Developers only need to make changes in the one program, which allows for faster development and reduces the likelihood of bugs being introduced. The work on reorganising and encapsulating the source code has also contributed to making the system easier to develop.

Although there are a number of companies developing speech recognition components, none that we investigated provided the functionality or worked the way we required for AuTM. As we only need to use a small portion of the functionality provided by the Microsoft API it would probably be best if we developed our own components in house. This would then increase the flexibility of AuTM allowing it to use multiple ASRs.

The Graphical User Interface was redesigned with increasing the usability in mind. The old GUI, although very flexible, proved to be difficult and confusing to use. The new GUI is easier to use and presents like a standard windows application. This new GUI will need to be further assessed in future user evaluation sessions.

Including streaming video and audio in AuTM increases its usability in a distributed operating mode where participants are in different locations. After experiments with various codecs to compress video data across the network *Morgan version 5.0 MPEG 3 codec* provided the best results. Due to the cost of this component it has not yet been included in AuTM. We have been utilising a standard codec that is packaged with the web cameras we are using. This provides a satisfactory result for demonstration purposes but we would envisage including the Morgan codec in future production versions of AuTM.

The audio splitting issue was resolved with the introduction of NS version 6.1 and the subsequent SDK. Although this solution does allow us to get a recording of each participant's utterance, it is difficult to manipulate this recording because it is in the NS proprietary .dra format. This reduces the flexibility of the system and makes using multiple ASRs more difficult. Ideally AuTM could record each utterance as a wave file that can be easily manipulated and played back on any computer running the windows

operating system. It would be useful to further investigate the use of VAC or WaveClone with AuTM as these components are further developed.

In considering improvements to AuTM's usability we wanted users not to have to worry about knowing computers' IP addresses. Unfortunately automating the connection process between computers in an AuTM session proved difficult. Because of the huge numbers of IP addresses involved in an open LAN it is not possible to automate the connection process. Participants must enter an IP address for the computer running as the moderator. In a closed LAN environment this process can be automated with AuTM statically assigning IP addresses. This will be included in future work on AuTM.

By upgrading the socket components used for connecting computers in an AuTM session the system has become more flexible and robust. Audio files can now be sent immediately with the utterances to the moderator. This allows participants to leave the session early and rejoin later if required. This increases the robustness of the system as an accurate transcription can still be created even if the connection between computers is lost. The new sockets also allow AuTM to operate over a wireless LAN again increasing its flexibility.

## 6. Future Work

### 6.1 Evaluation of the AuTM Prototype

The main goal of this work would be to evaluate the AuTM system in support of a pervasive meeting room environment. There has, so far, been no attempt to use the proposed system in a meeting or interview environment in order to assess its performance. A thorough evaluation of the prototype would enable us to identify the strengths and weaknesses of AuTM in order to improve it.

There are a number of key research areas relating to the AuTM system that can have a substantial impact on the way in which military personnel use information for decision-making in a command environment. It is envisaged that research results that would be generated from the proposed activities could shape the processes, architectures, and technologies to be employed in the new co-located headquarters and DJFHQ. Contributions from the research would also support further evolution of the Joint Command Systems Environment (JCSE) and help define systems requirements for coalition operations.

Specifically, the AuTM system has significant application towards:

- *Human Information Interaction* - control and multi-modal interaction including speech and dialog interfaces,

- *Future Workspace Environments* – architectures and infrastructures for enterprise enabled ubiquitous workspaces, virtual presence, and symbiotic workspaces.

It is envisaged that the work could be broken down into three stages, producing a research report.

*Stage 1 : Comparative Performance Evaluation of AuTM as a stand-alone system*

This stage would look at speed and accuracy of the meeting minutes creation process as the performance metrics, in comparison to similar COTS CSCW products that run a speech recogniser at the same time. The two main CSCW products of interest include Lotus SameTime and Microsoft's NetMeeting. The speech recognisers of interest would be the latest version of Dragon NaturallySpeaking, Microsoft Recognition Engine and IBM ViaVoice.

*Stage 2: Evaluation of the AuTM system in a task driven environment*

The goal of this stage would be to determine if AuTM offers any effective improvement of speed and accuracy over conventional means of recording proceedings in a task-driven meeting environment. Attention would also be paid to user perception and ease. AuTM will be evaluated in several meeting room environments; these include:

- Virtual meeting room environments where speakers in a meeting are remotely located, and
- Local meeting room environments where speakers are all located in the same meeting room.

*Stage 3 Evaluation of the Infrastructure associated with the AuTM system*

This stage would look at different microphone arrangements within a military meeting room environment. Microphones could include lapel, Bluetooth, goose-neck, boundary, array and infrared microphones. Each microphone could be evaluated specifically for the meeting record creation process in order to determine its impact on the meeting. Speaker separation techniques applied to both single microphones and multiple microphones would be employed to remove interference and cross-talk. Means of concurrently transcribing from multiple speakers using a speech recogniser could also be investigated.

This stage could also look at comparing and contrasting speech recognition based transcription systems versus manual and video-based means of meeting capture using a variety of software and hardware

## **6.2 Video display and Recording**

Concurrent video display has the main advantage of being less obtrusive than single panel display. In the current design the user is in charge of managing whom he/she wishes to view.

A more impressive solution would be to automatically detect voice and display the video linking to that particular speaker. In casual discussion-type dialog interruptions are quite frequent (Morgan et al., 2001), so the automated single-panel display would need to address this issue.

The recording of video is another development area of AuTM, which could serve as an important function for intelligence gathering environments. Recorded video could be logically indexed with recognised utterances that are time stamped within a meeting. This would provide greater support for decisions that were made through the assessment of non-verbal communication. Video For Windows provides an interface for the storage and manipulation of video, which would make this area of work very feasible to achieve.

## **6.3 Improving Connectivity**

This will include the research work on automatically finding and assigning IP addresses at the beginning of an AuTM session.

## **6.4 Use of Third Party Messaging Systems**

During the AuTM project some preliminary research was conducted on incorporating a messaging system in AuTM that does not rely on socket connections between two computers but uses a messaging subscription service. Appendix C is a briefing paper on the Elvin system developed by the DSTC and describes the system including the advantages and disadvantages of its use in the AuTM project. The integration of Elvin with AuTM has been taken up as a student project for 2003 in conjunction with the University of South Australia.

## **6.5 AuTM on Multiple Platforms**

AuTM has been designed and built using the Microsoft Windows 2000 operating system on both PC and laptop computers. This has been ideal for a concept demonstrator within the DSTO environment but we consider it to be too inflexible for a real system. AuTM would be made more useful and flexible if it could operate on a variety of different operating systems and platforms. There are a number of constraints with this idea, which are mainly due to the limited number of COTS ASRs

available on different platforms. The ASRs we have been using and researching run on most versions of Microsoft Windows but we have not found a COTS ASR that will run on the Unix/Linux platform and more research needs to be done to find a suitable ASR.

The first step in the task of enabling AuTM to run on multiple platforms is to have it operating on all versions of Microsoft Windows. This could include different devices such as Personal Digital Assistants (PDAs) that run Windows CE. Research and development in this direction will depend on client needs.

## **6.6 Transcript Storage and Retrieval**

Currently the end result of an AuTM meeting is the production of a transcript that can be saved as either a HTML document or a MS Word document. This allows for easy storage and transmitting over any network. The HTML document can then be displayed in any Web browser and could therefore be served up to users as a document on an Internet or Intranet site. A disadvantage with this format is that it is not easily editable. The MS Word version is easily editable but requires the user have a copy of MS Word to be able to view the transcript. Both these transcript formats have limitations.

A more ideal result is to have the transcript stored in a database that can be accessed by searching via a number of key words such as the date of the meeting, the names of the meeting participants or the title of the meeting. AuTM could allow the meeting moderator to save the transcript to a database for future retrieval.

## **6.7 Language Translation**

As the AuTM project was progressing during 2002 a separate project was undertaken developing a Language Translation Interface (LTI) (Biggs, 2003). The LTI is a concept demonstrator for the coordination of language translation computing resources and applications that must otherwise be accessed independently. The aim of the LTI is to provide users with translation results from a range of translation engines, thereby allowing them the possibility of obtaining the best available translation.

Integration of the LTI and AuTM allows meetings to be conducted across languages: a Cross-Language AuTM Session. A number of issues that may complicate this integration and the use of an AuTM system providing language translation has been identified by Biggs (2003).

It is unclear, at this stage, whether having AuTM provide real time translation facilities is going to be practical in a real meeting environment. This will be assessed and evaluated when the integration with AuTM 1.5 is complete in 2003.

## 7. Acknowledgements

We would like to acknowledge the assistance of several individuals who have contributed to this project. We would like to thank the Task Manager, Dr Ahmad Hashemi-Sakhatsari, who instigated the project and kept it going. We would also like to thank Andrew Zschorn for writing much of the original code. Many thanks go to Jason Littlefield who supervised the project from mid 2002, which helped the project stay focused on priorities. Our thanks also go to the following students who worked on the project during 2002 and 2003. Malcolm Blaney worked on integrating version 6 of NS so that the audio could be automatically saved by the speech recogniser and researched the use of Microsoft SAPI with AuTM. Mark Cook researched the area of trying to automatically find the IP addresses of participants' computers. Aik Kong worked on upgrading the TCP/IP sockets and integrating Microsoft's speech recognition engine.

## 8. References

- Biggs, J., 2003, *Language Translation Tools for the ADO: Investigation of language translation technology, and development and integration of a demonstration application*, DSTO Unpublished report, Edinburg, SA.
- Blaney, M., 2002, *AuTM and Microsofts Speech API*, unpublished paper, DSTO, Edinburg, SA.
- Chiu, P., Kapuskar, A., Reitmeier, S. and Wilcox, L. (1999): NoteLook: Taking notes in meetings with digital video and ink. *Proceedings of ACM Multimedia '99*, 149-158, ACM Press.
- Chiu, P., Boreczky, J., Girgensohn, A. and Kimber, D. (2001): LiteMinutes: An Internet-Based System for Multimedia Meeting Minutes. *Proceedings of World Wide Web 2001*, 140-149, ACM Press.
- Colbath, S. and Kubala, F. (1998): Rough'n'Ready: A Meeting Recorder and Browser. *A research note of the Perceptual User Interfaces Conference*, San Francisco, CA, November 1998.
- Dictaphone Corporation (2002), Enterprise Express EXSpeech,. [Accessed online 2 Sept. 2002], URL:<http://www.dictaphone.com/products/enterpriseexpress/exspeech/>.
- Gross, R., Bett, M., Yu, H., Zhu, X., Pan, Y., Yang, J. and Waibel, A. (2000): Towards a Multimodal Meeting Record. *Proceedings of the 2000 IEEE International Conference on Multimedia and Expo*, vol. 3, 2000, 1593-1596.

- Hashemi-Sakhtsari, A. and Littlefield, J. S. (2000) Project Proposal for a Speech-to-text Transcriber for Computer Supported Collaborative Work, Defence Science and Technology Organisation, Edinburgh.
- IBM Corporation (1) 2002, 'Introducing the IBM WebSphere Voice Server for Transcription', An IBM White Paper [Accessed online, 2 Sept. 2002], URL: [http://publib.boulder.ibm.com/voice/pdfs/white\\_papers/WSVSforTranscription.pdf](http://publib.boulder.ibm.com/voice/pdfs/white_papers/WSVSforTranscription.pdf).
- IBM Corporation (2) 2002, Lotus Sametime, [Accessed online, March 2002], URL: <http://www.lotus.com/products/lotussametime.nsf/wdocs/homepage.htm>
- Janin, A. (2001): Meeting Recorder. Avios, San Jose.
- Krishnamoorthy, B. (2000) *Automated Text Extraction of Meetings* Final, Unpublished Report, University of Adelaide and DSTO, Edinburgh.
- Kuhnen, R., Larossa-Greene, C. and Howes, S. L. (1999): *Distributed speech recognition system with multi-user input stations*, US Patent 6 308 158.
- Landay, J. A. and Davis, R. C (1999): *Making sharing pervasive: ubiquitous computing for shared note taking*. IBM Systems Journal, 38(4): 531-550, IBM Corp.
- Mehrabian, A. (1971). *Silent messages*. Belmont, CA: Wadsworth.
- Morgan, N., Baron, D., Edwards, J., Ellis, D., Gelbart, D., Janin, A. Pfau, T., Shriberg, E. and Stolcke, A. (2001): The Meeting Project at ICSI, *Proceedings of the Human Language Technology Conference*, in press.
- Muzychenko, E. (1) Virtual Audio Cable Home Page [Accessed online 4 April 2002] URL: <http://spider.nstu.nsk.su/music/software/eng/vac.html>
- Muzychenko, E., (2) *WaveClone Home Page* [Accessed online 4 April 2002], URL: <http://spider.nrcde.ru/music/software/eng/waveclone.html>
- Nazikian, F and Omoto, K (2002), *Listening in Context and Improvement of Proficiency*, in Proceedings of The 14th Central Association of Teachers of Japanese Conference [Accessed online 26 March 2003], URL: <http://polyglot.lss.wisc.edu/easian/CATJProceedings/11NazikianOmoto.pdf>
- Nunamaker, J. F., Dennis, A. R., Valacich, J. S., Vogel, D. R. and George, J. F. (1991): Electronic Meeting Systems to Support Group Work. *Communications of the ACM*, 34(7): 40-61, ACM Press.
- Oh, A., Tuchinda, R. and Wu, L. (2001): MeetingManager: A Collaborative Tool in the Intelligent Room. Proceedings of the Student Oxygen Workshop 2000, Cambridge, MA, USA.
- Scansoft, 2002,: *Dragon Naturally Speaking Professional Solutions Data Sheet*. [Accessed on line 30 August 2002,] URL: <FTP://ftp.scansoft.com/pub/doc/naturallspeaking/NS6ProfDatasheet.pdf>.

- Waibel, A., Bett, M., Finke, M. and Stiefelhagen, R. (1998): Meeting Browser: Tracking and Summarizing Meetings. *Proceedings of DARPA Broadcast News Transcription and Understanding Workshop*, 281-286, Lansdowne, VA, 1998.
- Waibel, A., Bett, M., Metze, F., Ries, K., Schaaf, T., Schoultz T., Soltan, H., Yu, H. and Zechner, K. (2001): Advances in Automatic Meeting Record Creation and Access. *Proceeding of the International Conference on Acoustics, Speech and Signal Processing* 2001.
- Yu, H., Clark, C., Malkin, R. and Waibel, A. (1998): Experiments in Automatic Meeting Transcription Using JRTk. *Proceedings of The 1998 IEEE International Conference of Acoustics, Speech and Signal Processing*, vol. 2, 921-924.
- Yu, H., Finke, M. and Waibel, A. (1999): Progress in Automatic Meeting Transcription. *Proceedings of 6th European Conference on Speech Communication and Technology (Eurospeech-99)*, Budapest, Volume 2: 695-698.
- Yu, H., Tomokiyo, T., Wang, Z. and Waibel, A. (2000): New Developments in Automatic Meeting Transcription. *Proceedings of the ICSLP*, Beijing, China, October 2000.
- Zschorn, A., Littlefield, J., Broughton, M., Dwyer, B., Hashemi-Sakhtsari, A. (2003) Transcription of Multiple Speakers using a Speaker-Dependent Speech Recogniser, DSTO Technical Report (DSTO-TR-xxxx), C2D, DSTO Edinburgh SA, being refereed.

## Appendix A: Assessment of Video Codecs

### A.1. Aim:

To optimise the speed and quality of video streaming, using analysis of various compression techniques.

### A.2. Background:

The goal of data compression is to represent an information source (i.e. a video signal) as accurately as possible using the fewest number of bits.

### A.3. Method:

Stage 1 prototype initially prompts the user to enter a compression technique from the list of 12. This compression is then used to stream video data remotely across the network. The size of the image and packet size is displayed before and after compression (see yet to be published screen shot). The figures were recorded for each compression technique.

A rating scale was then assigned to each compression technique based on the following characteristics:

Rating 1: Compression ratio <1:30 and picture quality excellent.

Rating 2: Compression ratio <1:25 and picture quality good.

Rating 3: Compression ratio <1:10 and picture quality average.

### A.4. Results:

Compression: *Morgan version 5.0 MPEG 3 codec*

Video Capture size: 230 KB (Before compression)

6 KB (After compression)

**Compression ratio: 1:38.3**

Most PC video capture and editing systems capture video using Motion JPEG video compression. In motion JPEG, each video frame is compressed separately using JPEG still image compression standard. No frame differencing or motion estimation is used to compress the images. This makes frame accurate editing without any loss of image quality during the editing possible.

**Rating: 1**

Compression: *Indeo Video 5.10* used 'Quick compress' option

Video Capture size: 230 KB (Before compression)

6 KB (After compression)

**Compression ratio: 1:38.3**

Research indicates that Indeo video codec is slightly more CPU intensive than the other codecs, however this is not obvious with AuTM.

**Rating: 2**

Compression: *Logitech Video*

Video Capture size: 230 KB (Before Compression)

115 KB (After Compression)

**Compression ratio: 1:2**

**Rating: 4**

Compression: *Cinepak codec by Radius*

Video capture size: 230 KB

10 KB (After compression)

**Compression ratio: 1:23**

Huge delay present through compression/decompression.

Picture quality relatively good. Cinepak reportedly provides the fastest playback of video; there is no evidence as such.

**Rating: 3**

Compression: *Intel Indeo Video R3.2*. Does not function

**Rating: N/a**

Compression: *Microsoft RLE*. Does not function

**Rating: N/a**

Compression: *Microsoft Video 1*

Video capture size: 230 KB (Uncompressed)

10 KB (Compressed)

**Compression ratio: 1:23**

The picture quality of this compression is not recommended (Very poor)

**Rating: 3**

Compression: *Microsoft H.263 Video Codec* -Does not function

**Rating: N/a**

Compression: *Microsoft H.261 Video Codec* -Does not function

**Rating: N/a**

Compression: *Logitech Video (1420)*

Video capture size: 230 KB (Uncompressed)

115 KB (Compressed)

**Compression ratio: 1:2**

**Rating: N/a**

The quality of this compression is not recommended.

Compression: *Intel 4:2:0 Video V2.50* –Does not function

Rating: N/a

Compression: *Logitech Video (YVU9)*

Video capture size: 230 KB (Uncompressed)

9 KB (Compressed)

Compression ratio: 1:2

Rating: N/a

The quality of this compression is not recommended.

## **Appendix B: Testing the Windows 98 version of WaveClone.**

### **B.1. Overview**

This experiment involved installing and tested the Windows 98 version of WaveClone on the Toshiba Tecra 8000 laptop to see if this application could provide a software solution to the audio splitting problem identified in the AuTM system..

### **B.2. Aims**

The aim of the experiment was to:

1. ensure the AuTM server and client programs would run on the Windows98 operating system using the audio splitting hardware connected to two sound cards.
2. run the AuTM server and client programs without the audio splitting hardware and produce the same results. The microphone would be plugged into just one sound card.

### **B.3. Methodology**

In the first part of the experiment the AuTM server and client were run. The client was initialized as usual using the Creative Vibra 128 device for DNS transcriptions and the Yamaha OPL3-SA device to record the utterances. This combination was successful in creating a transcript and a recording.

The second part of the experiment involved installing WaveClone and cloning the Wave in port of the Creative Vibra 128 audio device. I then ran the first part of the

experiment again to see if the installation process had any affect on AuTM's performance. No affect was noticeable.

The splitter hardware was then removed and the microphone was plugged directly into the Creative 128 audio device. When initializing the AuTM client the DNS software was set to the Vibra device and the AuTM client was set to record using the cloned device. This produced a successful transcription and an accurate recording but with some stammering. On repeating this test three further times the same results were obtained.

The next test involved initializing the AuTM client with the DNS software set to the cloned device and the AuTM client set to the Vibra device. This produced a successful transcript but failed to make a recording of the utterances. The .wav file was created but with nothing in it. The file's size was 0.

#### **B.4. Results**

WaveClone could provide a software solution to the audio splitting problem. The stammering in the recorded wave file did not reduce the sound quality to the extent of not being understood but it was not consistent in all tests. This stammering will need further investigation. Not being able to record from the original device, as in the last test, is a concern that needs further investigation.

## Appendix C: Briefing Paper on AuTM's Utilisation of Elvin

Author: Steve Graham

### C.1. Introduction

This briefing paper explores the possibility of using Elvin as a module with AuTM to provide the messaging subsystem between the client and server applications, and visa-versa. It gives a brief description of Elvin and AuTM then looks at some of the possible advantages and disadvantages of combining these applications into the one system.

### C.2. What is Elvin?

Elvin is a project located at and funded by the Distributed Systems Technology Centre. (DSTC) It is a messaging/notification system that allows a user to access a subscription service. Rather than messages being directed by applications from one component to another Elvin messages are routed to one or more locations, based entirely on the content of each message. As Elvin provides both flexibility and simplicity it can be used for communicating with components over the Internet.

Elvin requires both an Elvin Server and Elvin Client for the system to work. The Elvin4 server is written in portable C and runs on most Unix platforms and WindowsNT/2000. Command and Control Division (C2D) currently has an Elvin server available. Elvin clients can be written in a number of languages including C, C++, Java, Python, Perl, Palm and Emacs Lisp. I have requested information from DSTC regarding support for Delphi. It may be that AuTM would require an extra COM layer to connect it with Elvin.

An Elvin client sends a message in a prescribed, yet flexible, format that then gets broadcast to all other Elvin clients via the Elvin Server. Elvin uses a messaging protocol that uses name : value pairs. These name : value pairs can be read by a client application by parsing the message.. The Elvin client can be set up to only respond to certain messages depending on the content of the message.

### C.3. How could AuTM use Elvin?

We are researching the various architectural approaches that could be taken in developing AuTM further and exploring the idea of making AuTM more modular with the ability to plug in components. As can be seen from the above description of AuTM the messaging between client and server, and visa-versa, is a central feature of the system. A plug in component using Elvin could provide messaging services. AuTM currently uses its own messaging protocol. This protocol is fairly simple with certain

symbols identifying the type of message being sent, which is based on strings that are then parsed to extract parts that can then be used by the recipient.

#### **C.4. Possible Advantages**

As Elvin is a content based routing service no IP identification is required.

It would eliminate the need to use sockets for transferring messages between client computers.

The Elvin server provides a flexible security model. Currently AuTM provides no security in its messaging system. This is an issue when AuTM is used on an open LAN or the Internet.

Easily configured using server discovery, multiple processes can act together sharing subscriptions and notifications to minimise the load on system resources. This could allow us to add extra services to AuTM like file transfer and high quality audio and video.

While looking at the overall architecture for AuTM, in a team meeting in September, opinions were expressed about moving AuTM from a client/server architecture to a peer-to-peer architecture. Elvin could allow AuTM to move from a client/server architecture to a peer-to-peer architecture as it provides the broadcasting facilities currently provided by the AuTM server.

#### **C.5. Possible Disadvantages**

As Elvin requires a server to provide the routing service an Elvin server would have to be available for each AuTM meeting. This may not be desirable or possible in a closed LAN situation where a number of computers are connected together via a hub. If an Elvin server is introduced into this environment then set up and configuration issues will need to be considered. Thus introducing another component such as an Elvin server will complicate the AuTM system further.

AuTM's current messaging subsystem would require considerable rewriting of the source code to incorporate a component that utilized Elvin. This would need to be further assessed to ascertain a proper and realistic work-breakdown structure for the work.

It is unclear from the documentation on the web site the level and detail of the security provided. Security of messaging within AuTM is not an area that has yet been investigated, as it has been a low priority. It appears Elvin could provide a flexible level of security but it is unclear whether it will be sufficient for military applications.

Elvin may not readily support Delphi, the current IDE used to develop AuTM. This issue is being followed up with the Elvin website.

## DISTRIBUTION LIST

**An Automatic Transcriber of Meetings Utilising Speech Recognition Technology***Steve Graham and Jarrah Sladek***AUSTRALIA****DEFENCE ORGANISATION**

	No. of copies
<b>Task Sponsor</b>	
Comd DJFHQ	1
Deputy Chief Information Officer	1
<b>S&amp;T Program</b>	
Chief Defence Scientist	} shared copy
FAS Science Policy	
AS Science Corporate Management	
Director General Science Policy Development	
Counsellor Defence Science, London	Doc Data Sheet
Counsellor Defence Science, Washington	Doc Data Sheet
Scientific Adviser to MRDC, Thailand	Doc Data Sheet
Scientific Adviser Joint	1
Navy Scientific Adviser	1
Scientific Adviser - Army	1
Air Force Scientific Adviser	1
Scientific Adviser to the DMO M&A	Doc Data Sht & Dist List
Scientific Adviser to the DMO ELL	Doc Data Sht & Dist List
Director of Trials	1
<b>Information Sciences Laboratory</b>	
Chief of Command and Control Division	Doc Data Sheet
Research Leader Command & Intelligence Environments Branch	1
Research Leader Military Information Enterprise Branch	1
Research leader Theatre Command Analysis Branch	Doc Data Sheet
Head Virtual Enterprises	Doc Data Sheet
Head Systems Simulation and Assessment	Doc Data Sheet
Head Theatre Operations Analysis	Doc Data Sheet
Head Information Exploitation	Doc Data Sheet
Head Intelligence Analysis	Doc Data Sheet
Head C2 Australian Theatre	Doc Data Sheet
Head HQ Systems Experimentation	Doc Data Sheet
Head Information Systems	Doc Data Sheet

Head Human Systems Integration	1
Task Manager , JTW 01/091 Ahmad Hashemi-Sakhtsari, HSI Group	1
Author: Jarrah Sladek, HSI Group	1
Publications and Publicity Officer, C2D/EOC2D	1 shared copy
<b>DSTO Library and Archives</b>	
Library Edinburgh	2
Australian Archives	1
<b>Capability Systems Division</b>	
Director General Maritime Development	Doc Data Sheet
Director General Aerospace Development	Doc Data Sheet
Director General Information Capability	Doc Data Sheet
<b>Office of the Chief Information Officer</b>	
Deputy CIO	Doc Data Sheet
Director General Information Policy and Plans	Doc Data Sheet
AS Information Structures and Futures	Doc Data Sheet
AS Information Architecture and Management	Doc Data Sheet
Director General Australian Defence Simulation Office	Doc Data Sheet
<b>Strategy Group</b>	
Director General Military Strategy	Doc Data Sheet
Director General Preparedness	Doc Data Sheet
<b>HQAST</b>	
SO (Science) (ASJIC)	Doc Data Sheet
<b>Army</b>	
ABCA National Standardisation Officer, Land Warfare Development Sector,	
Puckapunyal	e-mailed Doc Data Sheet
SO (Science), Deployable Joint Force Headquarters (DJFHQ) (L), Enoggera QLD	
	Doc Data Sheet
SO (Science) - Land Headquarters (LHQ), Victoria Barracks NSW	
	Doc Data & Exec Summ
<b>Intelligence Program</b>	
DGSTA Defence Intelligence Organisation	1
Manager, Information Centre, Defence Intelligence Organisation	1
Assistant Secretary Corporate, Defence Imagery and Geospatial Organisation	
	Doc Data Sheet
<b>Defence Materiel Organisation</b>	
Head Airborne Surveillance and Control	Doc Data Sheet

Head Aerospace Systems Division	Doc Data Sheet
Head Electronic Systems Division	Doc Data Sheet
Head Maritime Systems Division	Doc Data Sheet
Head Land Systems Division	Doc Data Sheet
Head Industry Division	Doc Data Sheet
Chief Joint Logistics Command	Doc Data Sheet
Management Information Systems Division	Doc Data Sheet
Head Materiel Finance	Doc Data Sheet

**Defence Libraries**

Library Manager, DLS-Canberra	Doc Data Sheet
Library Manager, DLS - Sydney West	Doc Data Sheet

**OTHER ORGANISATIONS**

National Library of Australia	1
NASA (Canberra)	1

**UNIVERSITIES AND COLLEGES**

Australian Defence Force Academy	
Library	1
Head of Aerospace and Mechanical Engineering	1
Hargrave Library, Monash University	Doc Data Sheet
Librarian, Flinders University	1

**OUTSIDE AUSTRALIA****INTERNATIONAL DEFENCE INFORMATION CENTRES**

US Defense Technical Information Center	2
UK Defence Research Information Centre	2
Canada Defence Scientific Information Service	e-mail link to pdf
NZ Defence Information Centre	1

**ABSTRACTING AND INFORMATION ORGANISATIONS**

Library, Chemical Abstracts Reference Service	1
Engineering Societies Library, US	1
Materials Information, Cambridge Scientific Abstracts, US	1
Documents Librarian, The Center for Research Libraries, US	1

SPARES	5
--------	---

<b>Total number of copies:</b>	<b>38</b>
--------------------------------	-----------

<b>DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION DOCUMENT CONTROL DATA</b>				1. PRIVACY MARKING/CAVEAT (OF DOCUMENT)	
2. TITLE  An Automatic Transcriber of Meetings Utilising Speech Recognition Technology			3. SECURITY CLASSIFICATION (FOR UNCLASSIFIED REPORTS THAT ARE LIMITED RELEASE USE (L) NEXT TO DOCUMENT CLASSIFICATION)  Document (U) Title (U) Abstract (U)		
4. AUTHOR(S)  Steve Graham and Jarrah Sladek			5. CORPORATE AUTHOR  Information Sciences Laboratory PO Box 1500 Edinburgh South Australia 5111 Australia		
6a. DSTO NUMBER DSTO-CR-0355		6b. AR NUMBER AR-013-054		6c. TYPE OF REPORT Client Report	
				7. DOCUMENT DATE March 2004	
8. FILE NUMBER E 9505-25-210		9. TASK NUMBER JTW 01-092		11. NO. OF PAGES 46	
		10. TASK SPONSOR COMD DJFHQ DCIO		12. NO. OF REFERENCES 28	
13. URL on the World Wide Web  <a href="http://www.dsto.defence.gov.au/corporate/reports/DSTO-CR-0355.pdf">http://www.dsto.defence.gov.au/corporate/reports/DSTO-CR-0355.pdf</a>				14. RELEASE AUTHORITY  Chief, Command and Control Division	
15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT  <i>Approved for public release</i>					
OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SA 5111					
16. DELIBERATE ANNOUNCEMENT  No Limitations					
17. CITATION IN OTHER DOCUMENTS Yes					
18. DEFTEST DESCRIPTORS  Voice data processing, Speech recognition, Meetings, Information exchange					
19. ABSTRACT Conducting meetings is an important part of the Australian Defence Organisation (ADO) activities. Meetings span a broad range of knowledge sharing and collaborative activities; examples are interviews, informal discussions, brain-storming sessions, video-conferences and presentations. Thus, looking at ways to improve the capture of information from these meetings is of interest to the ADO. This report presents progress towards the application of speech recognition technology to automatically produce text transcriptions of collaborative meetings. The report reviews the literature on automatic transcription systems and describes the progress on the research and development of a concept demonstrator named AuTM (Automatic Transcriber of Meetings).					



**Australian Government**

**Department of Defence**

**Defence Science and  
Technology Organisation**

INFORMATION SCIENCES LABORATORY  
PO BOX 1500, EDINBURGH, SOUTH AUSTRALIA 5111  
AUSTRALIA. TELEPHONE (08) 8259 5555